## AmmarServer

Generated by Doxygen 1.8.6

Thu Jul 17 2014 15:44:54

## **Contents**

1	Amn	narServ	er		1						
	1.1	Introdu	ction and	History	1						
	1.2	What I	sit?		2						
	1.3	Coding	Style		2						
	1.4	Future	Plans		2						
	1.5	Deploy	ment		3						
	1.6	Depen	dencies .		3						
2	Bug	List	List 5								
3	Data	Structi	ure Index		7						
	3.1	Data S	tructures		7						
4	File	Index			9						
	4.1	File Lis	st		9						
5	Data	ta Structure Documentation 13									
	5.1	AmmS	erver_Dyn	amicRequest Struct Reference	13						
		5.1.1	Detailed	Description	13						
		5.1.2	Field Doo	cumentation	13						
			5.1.2.1	clientID	13						
			5.1.2.2	compressedContent	13						
			5.1.2.3	compressedContentSize	13						
			5.1.2.4	content	13						
			5.1.2.5	contentSize	13						
			5.1.2.6	GET_request	14						
			5.1.2.7	GET_request_length	14						
			5.1.2.8	headerResponse	14						
			5.1.2.9	MAXcompressedContentSize	14						
			5.1.2.10	MAXcontentSize	14						
			5.1.2.11	POST_request	14						
			5.1.2.12	POST_request_length	14						
	5.2	AmmS	Server_Instance Struct Reference								

iv CONTENTS

	5.2.1	Detailed	Description	15
	5.2.2	Field Doo	cumentation	15
		5.2.2.1	cache	15
		5.2.2.2	cacheHashMap	15
		5.2.2.3	CLIENT_THREADS_STARTED	15
		5.2.2.4	CLIENT_THREADS_STOPPED	15
		5.2.2.5	clientList	15
		5.2.2.6	clientRequestHandlerOverrideContext	15
		5.2.2.7	files_open	15
		5.2.2.8	instanceName	15
		5.2.2.9	loaded_cache_items	15
		5.2.2.10	loaded_cache_items_Kbytes	15
		5.2.2.11	pause_server	15
		5.2.2.12	prespawn_jobs_finished	15
		5.2.2.13	prespawn_jobs_started	15
		5.2.2.14	prespawn_turn_to_serve	16
		5.2.2.15	prespawned_pool	16
		5.2.2.16	server_running	16
		5.2.2.17	server_thread_id	16
		5.2.2.18	serversock	16
		5.2.2.19	settings	16
		5.2.2.20	stop_server	16
		5.2.2.21	templates_root	16
		5.2.2.22	threads_pool	16
		5.2.2.23	webserver_root	16
5.3	AmmS	erver_Inst	ance_Settings Struct Reference	16
	5.3.1	Detailed	Description	16
	5.3.2	Field Doo	cumentation	16
		5.3.2.1	BASE64PASSWORD	16
		5.3.2.2	BINDING_PORT	16
		5.3.2.3	PASSWORD	16
		5.3.2.4	PASSWORD_PROTECTION	16
		5.3.2.5	USERNAME	17
5.4	AmmS	erver_Req	uestOverride_Context Struct Reference	17
	5.4.1	Detailed	Description	17
	5.4.2	Field Doo	cumentation	17
		5.4.2.1	request	17
		5.4.2.2	request_override_callback	17
		5.4.2.3	requestHeader	17
5.5	AmmS	erver_RH_	_Context Struct Reference	18

CONTENTS

	5.5.1	Detailed Description			
	5.5.2	Field Do	cumentation	18	
		5.5.2.1	callback_cooldown	18	
		5.5.2.2	callback_every_x_msec	18	
		5.5.2.3	dynamicRequestCallbackFunction	18	
		5.5.2.4	last_callback	18	
		5.5.2.5	requestContext	18	
		5.5.2.6	resource_name	19	
		5.5.2.7	RH_Scenario	19	
		5.5.2.8	web_root_path	19	
5.6	cache_	_item Strud	ct Reference	19	
	5.6.1	Detailed	Description	19	
	5.6.2	Field Doo	cumentation	20	
		5.6.2.1	compressedContent	20	
		5.6.2.2	compressedContentSize	20	
		5.6.2.3	content	20	
		5.6.2.4	contentSize	20	
		5.6.2.5	contentTypeID	20	
		5.6.2.6	doNOTCacheRule	20	
		5.6.2.7	dynamicRequest	20	
		5.6.2.8	dynamicRequestCallbackFunction	20	
		5.6.2.9	modification	20	
5.7	clientLi	stContext	Struct Reference	20	
	5.7.1	Detailed	Description	21	
	5.7.2	Field Doo	cumentation	21	
		5.7.2.1	userList	21	
5.8	fastStri	ngParser	Struct Reference	21	
	5.8.1	Detailed	Description	21	
	5.8.2	Field Do	cumentation	21	
		5.8.2.1	contents	21	
		5.8.2.2	functionName	22	
		5.8.2.3	longestStringLength	22	
		5.8.2.4	MAXstringsLoaded	22	
		5.8.2.5	shortestStringLength	22	
		5.8.2.6	stringsLoaded	22	
5.9	fspStrir	ng Struct F	Reference	22	
	5.9.1	Detailed	Description	22	
	5.9.2	Field Do	cumentation	22	
		5.9.2.1	str	22	
		5.9.2.2	strIDFriendly	22	

vi CONTENTS

		5.9.2.3	strLength	. 22
5.10	guard_	byte Struc	t Reference	. 22
	5.10.1	Field Doo	eumentation	. 23
		5.10.1.1	checksum	. 23
5.11	hashM	ap Struct F	Reference	. 23
	5.11.1	Detailed I	Description	. 23
	5.11.2	Field Doo	eumentation	. 23
		5.11.2.1	clearItemCallbackFunction	. 23
		5.11.2.2	curNumberOfEntries	. 23
		5.11.2.3	entries	. 23
		5.11.2.4	entryAllocationStep	. 24
		5.11.2.5	hm_addLock	. 24
		5.11.2.6	hm_fileLock	. 24
		5.11.2.7	maxNumberOfEntries	. 24
5.12	hashM	apEntry St	ruct Reference	. 24
	5.12.1	Detailed I	Description	. 24
	5.12.2	Field Doo	sumentation	. 24
		5.12.2.1	hits	. 24
		5.12.2.2	key	. 24
		5.12.2.3	keyHash	. 24
		5.12.2.4	keyLength	. 24
		5.12.2.5	payload	. 24
		5.12.2.6	payloadLength	. 24
5.13	HTTPH	leader Stru	uct Reference	. 25
	5.13.1	Detailed I	Description	. 25
	5.13.2	Field Doo	umentation	. 25
		5.13.2.1	authorized	. 25
		5.13.2.2	boundary	. 25
		5.13.2.3	boundaryLength	. 25
		5.13.2.4	contentDisposition	. 26
		5.13.2.5	contentDispositionLength	. 26
		5.13.2.6	ContentLength	. 26
		5.13.2.7	contentType	. 26
		5.13.2.8	contentTypeLength	. 26
		5.13.2.9	cookie	. 26
		5.13.2.10	cookieLength	. 26
		5.13.2.11	eTag	. 26
		5.13.2.12	eTagLength	. 26
		5.13.2.13	GETquery	. 26
		5.13.2.14	headerRAW	. 26

CONTENTS vii

		5.13.2.15	headerRAWSize	26
		5.13.2.16	6 host	26
		5.13.2.17	hostLength	26
		5.13.2.18	B keepalive	26
		5.13.2.19	POSTrequest	26
		5.13.2.20	POSTrequestSize	26
		5.13.2.21	range_end	26
		5.13.2.22	2 range_start	26
		5.13.2.23	B referer	26
		5.13.2.24	FrefererLength	26
		5.13.2.25	5 requestType	26
		5.13.2.26	S resource	26
		5.13.2.27	supports_compression	26
		5.13.2.28	BuserAgent	26
		5.13.2.29	guserAgentLength	26
		5.13.2.30	verified_local_resource	26
5.14	HTTPT	ransaction	Struct Reference	27
	5.14.1	Detailed I	Description	27
	5.14.2	Field Doo	cumentation	27
		5.14.2.1	clientListID	27
		5.14.2.2	clientSock	27
		5.14.2.3	incomingHeader	28
		5.14.2.4	instance	28
		5.14.2.5	outgoingBody	28
		5.14.2.6	outgoingBodySize	28
		5.14.2.7	prespawnedThreadFlag	28
		5.14.2.8	resourceCacheID	28
		5.14.2.9	threadID	28
5.15	Image	Struct Refe	erence	28
	5.15.1	Field Doo	cumentation	28
		5.15.1.1	depth	28
		5.15.1.2	height	28
		5.15.1.3	imageSize	28
		5.15.1.4	pixels	28
		5.15.1.5	width	28
5.16	InputPa	arser Class	s Reference	28
	5.16.1	Construc	tor & Destructor Documentation	29
		5.16.1.1	InputParser	29
		5.16.1.2	~InputParser	29
	5.16.2	Member I	Function Documentation	29

viii CONTENTS

	5.16.2.1	DefaultDelimeterSetup	30
	5.16.2.2	GetDelimeter	30
	5.16.2.3	GetLowercaseWord	30
	5.16.2.4	GetUpcaseWord	30
	5.16.2.5	GetWord	31
	5.16.2.6	GetWordChar	31
	5.16.2.7	GetWordInt	31
	5.16.2.8	GetWordLength	31
	5.16.2.9	SeperateWords	32
	5.16.2.10	0 SeperateWordsCC	32
	5.16.2.11	1 SeperateWordsUC	32
	5.16.2.12	2 SetDelimeter	32
	5.16.2.13	3 Version	32
5.17 Input	ParserC Str	ruct Reference	33
5.17.	1 Field Do	ocumentation	33
	5.17.1.1	container_end	33
	5.17.1.2	container_start	33
	5.17.1.3	cur_container_count	33
	5.17.1.4	cur_delimeter_count	33
	5.17.1.5	delimeters	34
	5.17.1.6	3	
	5.17.1.7	guardbyte2	34
	5.17.1.8	guardbyte3	34
	5.17.1.9	guardbyte4	34
	5.17.1.10	0 local_allocation	34
		1 max_container_count	
		2 max_delimeter_count	
		3 str	
	5.17.1.14	4 str_length	34
		5 tokenlist	
		6 tokens_count	
		7 tokens_max	
		read Struct Reference	
		Description	
5.18.		ocumentation	
	5.18.2.1	client	
	5.18.2.2		
	5.18.2.3		
	5.18.2.4		
	5.18.2.5	keep_var_on_stack	36

CONTENTS

		5.18.2.6	port	36
		5.18.2.7	pre_spawned_thread	36
		5.18.2.8	thread_id	36
5.19	PassTo	PreSpawr	nedThread Struct Reference	36
	5.19.1	Field Doo	cumentation	36
		5.19.1.1	i_adapt	36
		5.19.1.2	instance	37
5.20	PreSpa	wnedThre	ead Struct Reference	37
	5.20.1	Detailed	Description	38
	5.20.2	Field Doo	cumentation	38
		5.20.2.1	busy	38
		5.20.2.2	client	38
		5.20.2.3	clientlen	38
		5.20.2.4	clientsock	38
		5.20.2.5	instance	38
		5.20.2.6	templates_root	38
		5.20.2.7	thread_id	38
		5.20.2.8	threadNum	38
		5.20.2.9	webserver_root	38
5.21	time_sr	nap Struct	Reference	38
	5.21.1	Field Doo	cumentation	38
		5.21.1.1	difference	38
5.22	timesta	mp Struct	Reference	38
	5.22.1	Detailed	Description	39
	5.22.2	Field Doo	cumentation	39
		5.22.2.1	day	39
		5.22.2.2	hour	39
		5.22.2.3	minute	39
		5.22.2.4	month	39
		5.22.2.5	second	39
		5.22.2.6	wday	39
		5.22.2.7	year	39
5.23	tokens	Struct Ref	ference	39
	5.23.1	Field Doo	cumentation	39
		5.23.1.1	length	39
		5.23.1.2	token_start	39
5.24	URLDE	Struct Re	eference	40
	5.24.1	Field Doo	cumentation	40
		5.24.1.1	longURL	40
		5.24.1.2	shortURL	40

CONTENTS

			5.24.1.3	shortURLHash	40
6	File	Docume	entation		41
	6.1	doc/Do	xygenMair	npage.h File Reference	41
	6.2	doc/he	lloworld.c l	File Reference	41
		6.2.1	Function	Documentation	42
			6.2.1.1	close_dynamic_content	42
			6.2.1.2	init_dynamic_content	42
			6.2.1.3	main	42
			6.2.1.4	prepare_helloworld_content_callback	43
		6.2.2	Variable I	Documentation	43
			6.2.2.1	helloworld	43
			6.2.2.2	helloworld_times_shown	43
			6.2.2.3	templates_root	43
			6.2.2.4	webserver_root	43
	6.3	src/Am	mCaptcha	/AmmCaptcha.h File Reference	43
		6.3.1	Function	Documentation	43
			6.3.1.1	AmmCaptcha_destroy	43
			6.3.1.2	AmmCaptcha_getCaptchaFrame	44
			6.3.1.3	AmmCaptcha_initialize	44
			6.3.1.4	AmmCaptcha_isReplyCorrect	44
			6.3.1.5	testAmmCaptcha	45
	6.4	src/Am	mCaptcha	/AmmCaptchaTester/main.c File Reference	45
		6.4.1	Function	Documentation	45
			6.4.1.1	main	46
	6.5	src/Am	mCaptcha	/main.c File Reference	46
		6.5.1	Macro De	efinition Documentation	47
			6.5.1.1	RANDOMIZE_AFTER_FAILED_ATTEMPT	47
		6.5.2	Function	Documentation	47
			6.5.2.1	AmmCaptcha_copyCaptchaJPEGImageWithCopy	47
			6.5.2.2	AmmCaptcha_destroy	47
			6.5.2.3	AmmCaptcha_getCaptchaFrame	48
			6.5.2.4	AmmCaptcha_initialize	48
			6.5.2.5	AmmCaptcha_isReplyCorrect	48
			6.5.2.6	AmmCaptcha_loadDictionary	49
			6.5.2.7	convertExternalIDToInternal	49
			6.5.2.8	RenderString	49
			6.5.2.9	testAmmCaptcha	50
		6.5.3	Variable I	Documentation	50
			6.5.3.1	captchaStrings	50

CONTENTS xi

		6.5.3.2	fontRAW	50
		6.5.3.3	fontX	50
		6.5.3.4	fontY	50
6.6	src/Am	mServerlib	o/InputParser/InputParser_C_Tester/main.c File Reference	50
	6.6.1	Macro De	efinition Documentation	51
		6.6.1.1	max_ret_word	51
	6.6.2	Function	Documentation	51
		6.6.2.1	IntermediateTests	51
		6.6.2.2	main	52
		6.6.2.3	ParseString	52
6.7	src/Am	nmServerlik	o/main.c File Reference	53
	6.7.1	Function	Documentation	55
		6.7.1.1	_FILES	55
		6.7.1.2	_GET	56
		6.7.1.3	_POST	56
		6.7.1.4	AmmServer_AddRequestHandler	56
		6.7.1.5	AmmServer_AddResourceHandler	57
		6.7.1.6	AmmServer_CheckIfHeaderBinaryAreTheSame	57
		6.7.1.7	AmmServer_DirectoryExists	58
		6.7.1.8	AmmServer_DoNOTCacheResource	58
		6.7.1.9	AmmServer_DoNOTCacheResourceHandler	58
		6.7.1.10	AmmServer_EraseFile	59
		6.7.1.11	AmmServer_Error	59
		6.7.1.12	AmmServer_ExecuteCommandLine	59
		6.7.1.13	AmmServer_FileExists	60
		6.7.1.14	AmmServer_FILES	60
		6.7.1.15	AmmServer_GeneralPrint	60
		6.7.1.16	AmmServer_GETArg	61
		6.7.1.17	AmmServer_GetInfo	62
		6.7.1.18	AmmServer_GetIntSettingValue	62
		6.7.1.19	AmmServer_GetStrSettingValue	62
		6.7.1.20	AmmServer_GlobalTerminationHandler	63
		6.7.1.21	AmmServer_POSTArg	63
		6.7.1.22	AmmServer_PreCacheFile	63
		6.7.1.23	AmmServer_ReadFileToMemory	64
		6.7.1.24	AmmServer_RegisterTerminationSignal	65
		6.7.1.25	AmmServer_RemoveResourceHandler	65
		6.7.1.26	AmmServer_ReplaceVarInMemoryFile	66
		6.7.1.27	AmmServer_Running	66
		6.7.1.28	AmmServer_SaveDynamicRequest	67

xii CONTENTS

		6.7.1.29	AmmServer_SelfCheck	68
		6.7.1.30	AmmServer_SetIntSettingValue	68
		6.7.1.31	AmmServer_SetStrSettingValue	68
		6.7.1.32	AmmServer_SignalCountAsBadClientBehaviour	69
		6.7.1.33	AmmServer_Start	69
		6.7.1.34	AmmServer_StartAdminInstance	70
		6.7.1.35	AmmServer_StartWithArgs	70
		6.7.1.36	AmmServer_Stop	71
		6.7.1.37	AmmServer_StringIsHTMLSafe	72
		6.7.1.38	AmmServer_Success	72
		6.7.1.39	AmmServer_Version	72
		6.7.1.40	AmmServer_Warning	72
		6.7.1.41	AmmServer_WriteFileFromMemory	74
	6.7.2	Variable I	Documentation	74
		6.7.2.1	TerminationCallback	74
6.8	src/Ser	vices/Amr	marServer/main.c File Reference	74
	6.8.1	Macro De	efinition Documentation	76
		6.8.1.1	ADMIN_BINDING_PORT	76
		6.8.1.2	DEFAULT_BINDING_PORT	76
		6.8.1.3	ENABLE_ADMIN_PAGE	76
		6.8.1.4	ENABLE_CHAT_BOX	76
		6.8.1.5	ENABLE_PASSWORD_PROTECTION	76
		6.8.1.6	MAX_BINDING_PORT	76
		6.8.1.7	MAX_SCRIPT_RESPONSE_SIZE	76
	6.8.2	Function	Documentation	76
		6.8.2.1	close_dynamic_content	76
		6.8.2.2	executeScriptFunction	76
		6.8.2.3	init_dynamic_content	77
		6.8.2.4	main	77
		6.8.2.5	prepare_chatbox_content_callback	78
		6.8.2.6	prepare_form_content_callback	78
		6.8.2.7	prepare_gps_content_callback	78
		6.8.2.8	prepare_random_content_callback	78
		6.8.2.9	prepare_stats_content_callback	78
		6.8.2.10	request_override_callback	78
	6.8.3	Variable I	Documentation	78
		6.8.3.1	admin_root	78
		6.8.3.2	admin_server	78
		6.8.3.3	chatbox	78
		6.8.3.4	default_server	78

CONTENTS xiii

		6.8.3.5	executeScript	79
		6.8.3.6	executeScriptRC	79
		6.8.3.7	form	79
		6.8.3.8	GET_override	79
		6.8.3.9	gps	79
		6.8.3.10	random_chars	79
		6.8.3.11	stats	79
		6.8.3.12	templates_root	79
		6.8.3.13	webserver_root	79
6.9	src/Ser	vices/Geo	PosShare/main.c File Reference	79
	6.9.1	Macro De	efinition Documentation	80
		6.9.1.1	ADMIN_BINDING_PORT	80
		6.9.1.2	DEFAULT_BINDING_PORT	80
		6.9.1.3	ENABLE_ADMIN_PAGE	80
		6.9.1.4	MAX_BINDING_PORT	80
	6.9.2	Function	Documentation	80
		6.9.2.1	close_dynamic_content	80
		6.9.2.2	init_dynamic_content	80
		6.9.2.3	main	81
		6.9.2.4	prepare_gps_content_callback	81
		6.9.2.5	request_override_callback	81
	6.9.3	Variable I	Documentation	81
		6.9.3.1	admin_root	81
		6.9.3.2	default_server	82
		6.9.3.3	GET_override	82
		6.9.3.4	gps	82
		6.9.3.5	templates_root	82
		6.9.3.6	webserver_root	82
6.10	src/Ser	vices/MyL	oader/main.c File Reference	82
	6.10.1	Macro De	efinition Documentation	83
		6.10.1.1	DEFAULT_BINDING_PORT	83
	6.10.2	Function	Documentation	83
		6.10.2.1	close_dynamic_content	83
		6.10.2.2	init_dynamic_content	83
		6.10.2.3	main	83
		6.10.2.4	prepare_stats_content_callback	84
		6.10.2.5	processUploadCallback	84
		6.10.2.6	request_override_callback	85
	6.10.3	Variable I	Documentation	85
		6.10.3.1	default_server	85

XIV

6.10.3.	2 GET_override
6.10.3.	3 stats
6.10.3.	4 templates_root
6.10.3.	5 uploadProcessor
6.10.3.	6 webserver_root
6.11 src/Services/M	yURL/main.c File Reference
6.11.1 Macro	Definition Documentation
6.11.1.	1 DEFAULT_BINDING_PORT
6.11.1.	2 DYNAMIC_PAGES_MEMORY_COMMITED
6.11.1.	3 ENABLE_CAPTCHA_SYSTEM
6.11.1.	4 LINK_ALLOCATION_STEP
6.11.1.	5 MAX_BINDING_PORT 87
6.11.1.	6 MAX_CAPTCHA_JPG_SIZE
6.11.1.	7 MAX_LINKS
6.11.1.	8 MAX_LONG_URL_SIZE
6.11.1.	9 MAX_TO_SIZE
6.11.1.	10 REGROUP_AFTER_X_UNSORTED_LINKS
6.11.1.	11 USE_BINARY_SEARCH
6.11.2 Function	on Documentation
6.11.2.	1 Add_MyURL
6.11.2.	2 allocateLinksIfNeeded
6.11.2.	3 Append2MyURLDBFile 88
6.11.2.	4 close_dynamic_content
6.11.2.	5 Find_longURL
6.11.2.	6 Find_longURLSerial
6.11.2.	7 Get_longURL
6.11.2.	8 hashURL
6.11.2.	9 init_dynamic_content
6.11.2.	10 is_an_unsafe_str
6.11.2.	11 isURLDBSorted
6.11.2.	12 LoadMyURLDBFile
6.11.2.	13 main
6.11.2.	14 printURLDB
6.11.2.	15 resolveRequest
6.11.2.	16 ResortDB
6.11.2.	17 ReWriteMyURLDBFile
6.11.2.	18 serve_captcha_page
6.11.2.	19 serve_create_url_page
6.11.2.	20 serve_error_url_page
6.11.2.	21 serve_goto_url_page

CONTENTS xv

	6.11.2.22 struct_cmp_urldb_items	94
6.11.3	Variable Documentation	94
	6.11.3.1 allocated_links	94
	6.11.3.2 captcha_url	94
	6.11.3.3 create_url	94
	6.11.3.4 db_addIDLock	94
	6.11.3.5 db_file	94
	6.11.3.6 db_fileLock	94
	6.11.3.7 default_failed	94
	6.11.3.8 error_url	94
	6.11.3.9 goto_url	94
	6.11.3.10 indexPage	94
	6.11.3.11 indexPageLength	94
	6.11.3.12 indexPagePath	94
	6.11.3.13 links	95
	6.11.3.14 loaded_links	95
	6.11.3.15 myurl_server	95
	6.11.3.16 requestResolver	95
	6.11.3.17 service_filename	95
	6.11.3.18 service_filename_noslash	95
	6.11.3.19 service_root	95
	6.11.3.20 service_root_withoutfilename	95
	6.11.3.21 sorted_links	95
	6.11.3.22 templates_root	95
	6.11.3.23 webserver_root	95
6.12 src/Ser	vices/SimpleTemplate/main.c File Reference	95
6.12.1	Macro Definition Documentation	96
	6.12.1.1 DEFAULT_BINDING_PORT	96
6.12.2	Function Documentation	96
	6.12.2.1 close_dynamic_content	96
	6.12.2.2 init_dynamic_content	96
	6.12.2.3 main	96
	6.12.2.4 prepare_random_content_callback	97
	6.12.2.5 prepare_stats_content_callback	97
	6.12.2.6 request_override_callback	97
6.12.3	Variable Documentation	97
	6.12.3.1 default_server	97
	6.12.3.2 GET_override	97
	6.12.3.3 random_chars	97
	6.12.3.4 stats	97

xvi CONTENTS

		6.12.3.5	templates_root	97
		6.12.3.6	webserver_root	98
6.13	src/Stri	ngRecogn	izer/main.c File Reference	98
	6.13.1	Function	Documentation	98
		6.13.1.1	main	98
6.14	src/Am	mCaptcha	/imaging.c File Reference	98
	6.14.1	Macro De	efinition Documentation	99
		6.14.1.1	DISPLAY_DEBUG_INFO	99
		6.14.1.2	PPMREADBUFLEN	99
		6.14.1.3	READ_CREATES_A_NEW_PIXEL_BUFFER	99
	6.14.2	Function	Documentation	99
		6.14.2.1	bitBltImage	99
		6.14.2.2	bitBltImageRotated	99
		6.14.2.3	copylmage	99
		6.14.2.4	createImage	00
		6.14.2.5	destroylmage	00
		6.14.2.6	ReadPPM	00
		6.14.2.7	WritePPM	00
6.15	src/Am	mCaptcha	/imaging.h File Reference	00
	6.15.1	Function	Documentation	00
		6.15.1.1	bitBltImage	00
		6.15.1.2	copyImage	00
		6.15.1.3	createImage	00
		6.15.1.4	destroyImage	00
		6.15.1.5	ReadPPM	00
		6.15.1.6	WritePPM	00
6.16	src/Am	mCaptcha	/img_warp.c File Reference	01
	6.16.1	Macro De	efinition Documentation	01
		6.16.1.1	ABS 1	01
		6.16.1.2	ABSDIFF	01
	6.16.2	Function	Documentation	01
		6.16.2.1	coolPHPWave	01
		6.16.2.2	warpImage	02
6.17	src/Am	mCaptcha	/img_warp.h File Reference	02
	6.17.1	Function	Documentation	03
		6.17.1.1	coolPHPWave	03
		6.17.1.2	warpImage	04
6.18	src/Am	mCaptcha	/jpgInput.c File Reference	04
	6.18.1	Function	Documentation	05
		6.18.1.1	empty_buffer	05

CONTENTS xvii

	6.18.1.2	fastJPGHeaderCheck	105
	6.18.1.3	init_buffer	105
	6.18.1.4	jpegtest	105
	6.18.1.5	ReadJPEG	105
	6.18.1.6	term_buffer	105
	6.18.1.7	WriteJPEGFile	106
	6.18.1.8	WriteJPEGInternal	106
	6.18.1.9	WriteJPEGMemory	107
6.19 src/A	mmCaptcha	a/jpgInput.h File Reference	107
6.19.	Macro De	efinition Documentation	108
	6.19.1.1	USE_JPG_FILES	108
6.19.2	2 Function	Documentation	108
	6.19.2.1	ReadJPEG	108
	6.19.2.2	WriteJPEGFile	108
	6.19.2.3	WriteJPEGMemory	109
6.20 src/Ai	mmServerlil	b/AmmServerlib.h File Reference	109
6.20.	I Detailed	Description	113
6.20.2	2 Macro De	efinition Documentation	113
	6.20.2.1	AMMAR_SERVER_HTTP_HEADER_SPEC	113
	6.20.2.2	MAX_FILE_PATH	113
	6.20.2.3	MAX_INSTANCE_NAME_STRING	113
	6.20.2.4	MAX_IP_STRING_SIZE	113
	6.20.2.5	MAX_QUERY	113
	6.20.2.6	MAX_RESOURCE	113
	6.20.2.7	POPEN_BUFFER_SIZE	113
6.20.3	B Enumera	tion Type Documentation	113
	6.20.3.1	AmmServInfos	113
	6.20.3.2	AmmServSettings	114
	6.20.3.3	AmmServStrSettings	114
	6.20.3.4	RHScenarios	114
	6.20.3.5	TypesOfRequests	114
6.20.4	Function	Documentation	115
	6.20.4.1	_FILES	115
	6.20.4.2	_GET	115
	6.20.4.3	_POST	115
	6.20.4.4	AmmServer_AddRequestHandler	115
	6.20.4.5	AmmServer_AddResourceHandler	116
	6.20.4.6	AmmServer_CheckIfHeaderBinaryAreTheSame	116
	6.20.4.7	AmmServer_DirectoryExists	117
	6.20.4.8	AmmServer_DoNOTCacheResource	117

xviii CONTENTS

		5.20.4.9 AmmServer_DonOT	CacheResourceHandler	 	 	117
		6.20.4.10 AmmServer_EraseFi	le	 	 	118
		6.20.4.11 AmmServer_Error .		 	 	118
		6.20.4.12 AmmServer_Execute	eCommandLine	 	 	118
		6.20.4.13 AmmServer_FileExis	ots	 	 	119
		6.20.4.14 AmmServer_FILES		 	 	119
		6.20.4.15 AmmServer_GETArg	]	 	 	119
		6.20.4.16 AmmServer_GetInfo		 	 	120
		6.20.4.17 AmmServer_GetIntS	ettingValue	 	 	120
		6.20.4.18 AmmServer_GetStrS	SettingValue	 	 	120
		6.20.4.19 AmmServer_POSTA	rg	 	 	121
		6.20.4.20 AmmServer_ReadFil	eToMemory	 	 	121
		6.20.4.21 AmmServer_Registe	rTerminationSignal	 	 	122
		6.20.4.22 AmmServer_Remover	eResourceHandler	 	 	123
		6.20.4.23 AmmServer_Replace	eVarInMemoryFile	 	 	123
		6.20.4.24 AmmServer_Running	g	 	 	124
		6.20.4.25 AmmServer_SaveDy	namicRequest	 	 	124
		6.20.4.26 AmmServer_SelfChe	eck	 	 	125
		6.20.4.27 AmmServer_SetIntS	ettingValue	 	 	126
		6.20.4.28 AmmServer_SetStrS	ettingValue	 	 	126
		6.20.4.29 AmmServer_SignalC	countAsBadClientBehaviour	 	 	126
		6.20.4.30 AmmServer_Start .		 	 	127
		6.20.4.31 AmmServer_StartAd	minInstance	 	 	127
		6.20.4.32 AmmServer_StartWi	thArgs	 	 	128
		6.20.4.33 AmmServer_Stop .		 	 	129
		6.20.4.34 AmmServer_StringIs	HTMLSafe	 	 	130
		6.20.4.35 AmmServer_Succes	s	 	 	130
		6.20.4.36 AmmServer_Version		 	 	131
		6.20.4.37 AmmServer_Warning	j	 	 	131
		6.20.4.38 AmmServer_WriteFil	eFromMemory	 	 	131
6.21	src/Am	Serverlib/cache/client_list.c Fil	e Reference	 	 	131
	6.21.1	Macro Definition Documentatio	n	 	 	132
		6.21.1.1 COMPILE_WITH_CL	LIENT_LIST	 	 	132
	6.21.2	Function Documentation		 	 	132
		6.21.2.1 clientList_close		 	 	133
		6.21.2.2 clientList_GetClientIc	1	 	 	134
		6.21.2.3 clientList_initialize .		 	 	134
		6.21.2.4 clientList_isClientAllo	wedToUseResource	 	 	134
		6.21.2.5 clientList_isClientBar	nned	 	 	134
		6.21.2.6 clientList_signalClier	tStoppedUsingResource	 	 	135

CONTENTS xix

6.22	src/Am	mServerlib	/cache/client_list.h File Reference	35
	6.22.1	Detailed [	Description	36
	6.22.2	Typedef D	ocumentation	36
		6.22.2.1	clientID	36
	6.22.3	Function I	Documentation	36
		6.22.3.1	clientList_close	36
		6.22.3.2	clientList_GetClientId	37
		6.22.3.3	clientList_initialize	37
		6.22.3.4	clientList_isClientAllowedToUseResource	37
		6.22.3.5	clientList_isClientBanned	37
		6.22.3.6	clientList_signalClientStoppedUsingResource	38
6.23	src/Am	mServerlib	/cache/dynamic_requests.c File Reference	38
	6.23.1	Function I	Documentation	39
		6.23.1.1	callClientRequestHandler	39
		6.23.1.2	dynamicRequest_ContentAvailiable	39
		6.23.1.3	dynamicRequest_serveContent	39
		6.23.1.4	saveDynamicRequest	40
6.24	src/Am	mServerlib	/cache/dynamic_requests.h File Reference	40
	6.24.1	Detailed [	Description	42
	6.24.2	Function I	Documentation	42
		6.24.2.1	callClientRequestHandler	42
		6.24.2.2	dynamicRequest_ContentAvailiable	42
		6.24.2.3	dynamicRequest_serveContent	42
		6.24.2.4	saveDynamicRequest	43
6.25	src/Am	mServerlib	/cache/file_caching.c File Reference	43
	6.25.1	Function I	Documentation	45
		6.25.1.1	cache_AddDoNOTCacheRuleForResource	45
		6.25.1.2	cache_AddFile	45
		6.25.1.3	cache_AddMemoryBlock	46
		6.25.1.4	cache_ChangeRequestIfTemplateRequested	46
		6.25.1.5	cache_CreateResource	47
		6.25.1.6	cache_Destroy	47
		6.25.1.7	cache_DestroyResource	47
		6.25.1.8	cache_FindResource	47
		6.25.1.9	cache_GetHashOfResource	48
		6.25.1.10	cache_GetResource	48
		6.25.1.11	cache_Initialize	49
		6.25.1.12	cache_LoadResourceFromDisk	50
		6.25.1.13	cache_RemoveContextAndResource	50
		6.25.1.14	cache_RemoveResource	51

CONTENTS

		6.25.1.15	cache_ResourceExists	1
		6.25.1.16	freeMallocIfNeeded	1
6.26	src/Am	mServerlib	/cache/file_caching.h File Reference	1
	6.26.1	Detailed [	Description	3
	6.26.2	Function	Documentation	3
		6.26.2.1	cache_AddDoNOTCacheRuleForResource	3
		6.26.2.2	cache_AddFile	4
		6.26.2.3	cache_AddMemoryBlock	4
		6.26.2.4	cache_ChangeRequestIfTemplateRequested	5
		6.26.2.5	cache_Destroy	5
		6.26.2.6	cache_FindResource	6
		6.26.2.7	cache_GetHashOfResource	6
		6.26.2.8	cache_GetResource	6
		6.26.2.9	cache_Initialize	7
		6.26.2.10	cache_RemoveContextAndResource	8
		6.26.2.11	cache_ResourceExists	8
		6.26.2.12	freeMallocIfNeeded	9
6.27	src/Am	mServerlib	/cache/file_compression.c File Reference	9
	6.27.1	Function	Documentation	0
		6.27.1.1	CreateCompressedVersionofCachedResource	0
		6.27.1.2	CreateCompressedVersionofDynamicContent	0
		6.27.1.3	CreateCompressedVersionofStaticContent	0
		6.27.1.4	CreateCompressedVersionofStaticContentPreloading	1
6.28	src/Am	mServerlib	/cache/file_compression.h File Reference	2
	6.28.1	Detailed [	Description	3
	6.28.2	Function	Documentation	3
		6.28.2.1	CreateCompressedVersionofDynamicContent	3
		6.28.2.2	CreateCompressedVersionofStaticContent	3
		6.28.2.3	CreateCompressedVersionofStaticContentPreloading	4
6.29	src/Am	mServerlib	/hashmap/hashmap.c File Reference	4
	6.29.1	Function	Documentation	6
		6.29.1.1	cmpHashTableItems	6
		6.29.1.2	hashFunction	6
		6.29.1.3	hashMap_Add	6
		6.29.1.4	hashMap_AddULong	7
		6.29.1.5	hashMap_Clear	7
		6.29.1.6	hashMap_ContainsKey	8
		6.29.1.7	hashMap_ContainsValue	8
		6.29.1.8	hashMap_Create	9
		6.29.1.9	hashMap_Destroy	9

CONTENTS xxi

	6.29.1.10 hashMap_FindIndex	170
	6.29.1.11 hashMap_GetCurrentNumberOfEntries	170
	6.29.1.12 hashMap_GetHashAtIndex	171
	6.29.1.13 hashMap_GetKeyAtIndex	171
	6.29.1.14 hashMap_GetMaxNumberOfEntries	172
	6.29.1.15 hashMap_GetPayload	172
	6.29.1.16 hashMap_GetULongPayload	173
	6.29.1.17 hashMap_Grow	173
	6.29.1.18 hashMap_IsOK	173
	6.29.1.19 hashMap_IsSorted	174
	6.29.1.20 hashMap_LoadToFile	174
	6.29.1.21 hashMap_SaveToFile	174
	6.29.1.22 hashMap_Sort	175
	6.29.1.23 hashmap_SwapRecords	175
6.30 src/A	AmmServerlib/hashmap/hashmap.h File Reference	176
6.30	.1 Detailed Description	177
6.30	.2 Function Documentation	178
	6.30.2.1 hashFunction	178
	6.30.2.2 hashMap_Add	178
	6.30.2.3 hashMap_AddULong	178
	6.30.2.4 hashMap_Clear	179
	6.30.2.5 hashMap_ContainsKey	179
	6.30.2.6 hashMap_ContainsValue	180
	6.30.2.7 hashMap_Create	180
	6.30.2.8 hashMap_Destroy	181
	6.30.2.9 hashMap_FindIndex	181
	6.30.2.10 hashMap_GetCurrentNumberOfEntries	182
	6.30.2.11 hashMap_GetHashAtIndex	182
	6.30.2.12 hashMap_GetKeyAtIndex	183
	6.30.2.13 hashMap_GetMaxNumberOfEntries	183
	6.30.2.14 hashMap_GetPayload	184
	6.30.2.15 hashMap_GetULongPayload	184
	6.30.2.16 hashMap_LoadToFile	185
	6.30.2.17 hashMap_SaveToFile	185
	6.30.2.18 hashMap_Sort	186
	6.30.2.19 hashmap_SwapRecords	186
6.31 src/A	AmmServerlib/header_analysis/http_header_analysis.c File Reference	187
6.31	.1 Macro Definition Documentation	188
	6.31.1.1 CR	188
	6.31.1.2 LF	188

xxii CONTENTS

	6.31.2	Function Documentation
		6.31.2.1 AnalyzeHTTPHeader
		6.31.2.2 AnalyzeHTTPLineRequest
		6.31.2.3 AppendPOSTRequestToHTTPHeader
		6.31.2.4 FreeHTTPHeader
		6.31.2.5 HTTPHeaderComplete
		6.31.2.6 HTTPHeaderIsPOST
		6.31.2.7 ProcessAuthorizationHTTPLine
		6.31.2.8 ProcessFirstHTTPLine
		6.31.2.9 ProcessRangeHTTPLine
		6.31.2.10 ReceiveHTTPHeader
6.32	src/Am	mServerlib/header_analysis/http_header_analysis.h File Reference
	6.32.1	Detailed Description
	6.32.2	Function Documentation
		6.32.2.1 AnalyzeHTTPHeader
		6.32.2.2 AppendPOSTRequestToHTTPHeader
		6.32.2.3 FreeHTTPHeader
		6.32.2.4 HTTPHeaderComplete
		6.32.2.5 HTTPHeaderIsPOST
		6.32.2.6 ReceiveHTTPHeader
6.33	src/Am	mServerlib/header_analysis/post_header_analysis.c File Reference
	6.33.1	Function Documentation
		6.33.1.1 AnalyzePOSTLineRequest
6.34	src/Am	mServerlib/header_analysis/post_header_analysis.h File Reference
	6.34.1	Detailed Description
	6.34.2	Function Documentation
		6.34.2.1 AnalyzePOSTLineRequest
6.35	src/Am	mServerlib/InputParser/InputParser.cpp File Reference
	6.35.1	Function Documentation
		6.35.1.1 Version
	6.35.2	Variable Documentation
		6.35.2.1 ver
6.36	src/Am	mServerlib/InputParser/InputParser.h File Reference
6.37	src/Am	mServerlib/InputParser/InputParser_C.c File Reference
	6.37.1	Macro Definition Documentation
		6.37.1.1 WARN_ABOUT_INCORRECTLY_ALLOCATED_STACK_STRINGS 203
	6.37.2	Function Documentation
		6.37.2.1 CheckDelimeterNumOk
		6.37.2.2 CheckIPCOk
		6.37.2.3 CheckWordNumOk

CONTENTS xxiii

		0.37.2.4	inputParser_ClearNonCharacters	204
		6.37.2.5 I	InputParser_Create	204
		6.37.2.6 I	InputParser_DefaultDelimeters	204
		6.37.2.7	InputParser_Destroy	204
		6.37.2.8	InputParser_GetDelimeter	205
		6.37.2.9	InputParser_GetLowercaseWord	205
		6.37.2.10	InputParser_GetUpcaseWord	205
		6.37.2.11	InputParser_GetWord	205
		6.37.2.12	InputParser_GetWordChar	206
		6.37.2.13	InputParser_GetWordFloat	206
		6.37.2.14	InputParser_GetWordInt	206
		6.37.2.15	InputParser_GetWordLength	206
		6.37.2.16	InputParser_SelfCheck	207
		6.37.2.17	InputParser_SeperateWords	207
		6.37.2.18	InputParser_SeperateWordsCC	207
		6.37.2.19	InputParser_SeperateWordsUC	207
		6.37.2.20	InputParser_SetDelimeter	208
		6.37.2.21	InputParser_TrimCharacters	208
		6.37.2.22	InputParser_TrimCharactersEnd	208
		6.37.2.23	InputParser_TrimCharactersStart	208
		6.37.2.24	InputParser_WordCompare	208
		6.37.2.25	InputParser_WordCompareAuto	209
		6.37.2.26	InputParser_WordCompareNoCase	209
		6.37.2.27	InputParser_WordCompareNoCaseAuto	209
		6.37.2.28	InputParserC_Version	209
		6.37.2.29	Str2Int_internal	209
	6.37.3	Variable D	ocumentation	209
		6.37.3.1	_ipc_ver	209
		6.37.3.2	warningsAboutIncorrectlyAllocatedStackIssued	209
6.38	src/Am	mServerlib/	InputParser/InputParser_C.h File Reference	209
	6.38.1	Macro Def	inition Documentation	211
		6.38.1.1	CONTAINERS_MAX	211
		6.38.1.2 I	DELIM_MAX_MAX	211
		6.38.1.3 I	MAX_COMPLICITY	211
		6.38.1.4 I	MAX_MEMORY	211
		6.38.1.5 I	MAX_STRING	211
	6.38.2	Function D	Documentation	211
		6.38.2.1	CheckWordNumOk	211
		6.38.2.2	InputParser_ClearNonCharacters	212
		6.38.2.3 I	InputParser_Create	212

xxiv CONTENTS

		6.38.2.4	InputParser_DefaultDelimeters	212
		6.38.2.5	InputParser_Destroy	212
		6.38.2.6	InputParser_GetDelimeter	212
		6.38.2.7	InputParser_GetLowercaseWord	213
		6.38.2.8	InputParser_GetUpcaseWord	213
		6.38.2.9	InputParser_GetWord	213
		6.38.2.10	InputParser_GetWordChar	213
		6.38.2.11	InputParser_GetWordFloat	214
		6.38.2.12	InputParser_GetWordInt	214
		6.38.2.13	InputParser_GetWordLength	214
		6.38.2.14	InputParser_SelfCheck	214
		6.38.2.15	InputParser_SeperateWords	215
		6.38.2.16	InputParser_SeperateWordsCC	215
		6.38.2.17	InputParser_SeperateWordsUC	215
		6.38.2.18	InputParser_SetDelimeter	215
		6.38.2.19	InputParser_TrimCharacters	216
		6.38.2.20	InputParser_TrimCharactersEnd	216
		6.38.2.21	InputParser_TrimCharactersStart	216
		6.38.2.22	InputParser_WordCompare	216
		6.38.2.23	InputParser_WordCompareAuto	216
		6.38.2.24	InputParser_WordCompareNoCase	216
		6.38.2.25	InputParser_WordCompareNoCaseAuto	217
		6.38.2.26	InputParserC_Version	217
6.39	src/Am	mServerlib	n/network/file_server.c File Reference	217
	6.39.1	Function	Documentation	218
		6.39.1.1	SendErrorFile	218
		6.39.1.2	SendFile	218
		6.39.1.3	SendMemoryBlockAsFile	219
		6.39.1.4	SendPart	220
		6.39.1.5	TransmitFileToSocket	220
		6.39.1.6	TransmitFileToSocketInternal	221
	6.39.2	Variable [	Documentation	221
		6.39.2.1	files_open	221
6.40	src/Am	mServerlib	n/network/file_server.h File Reference	221
	6.40.1	Detailed [	Description	223
	6.40.2	Function	Documentation	223
		6.40.2.1	SendErrorFile	223
		6.40.2.2	SendFile	223
		6.40.2.3	SendMemoryBlockAsFile	224
6.41	src/Am	mServerlib	/network/sendHTTPHeader.c File Reference	225

CONTENTS xxv

	6.41.1	Function	Documentation
		6.41.1.1	SendAuthorizationHeader
		6.41.1.2	SendErrorCodeHeader
		6.41.1.3	SendNotModifiedHeader
		6.41.1.4	SendSuccessCodeHeader
6.42	src/Am	mServerlib	o/network/sendHTTPHeader.h File Reference
	6.42.1	Detailed	Description
	6.42.2	Function	Documentation
		6.42.2.1	SendAuthorizationHeader
		6.42.2.2	SendErrorCodeHeader
		6.42.2.3	SendNotModifiedHeader
		6.42.2.4	SendSuccessCodeHeader
6.43	src/Am	mServerlik	o/server_configuration.c File Reference
	6.43.1	Function	Documentation
		6.43.1.1	AssignStr
		6.43.1.2	EmmitPossibleConfigurationWarnings
		6.43.1.3	instance_CountFreeOP
		6.43.1.4	instance_CountNewMallocOP
		6.43.1.5	instance_WeCanCommitMoreMemory
		6.43.1.6	LoadConfigurationFile
		6.43.1.7	SetUsernameAndPassword
	6.43.2	Variable I	Documentation
		6.43.2.1	AccessLog
		6.43.2.2	AccessLogEnable
		6.43.2.3	CACHING_ENABLED
		6.43.2.4	CHANGE_PRIORITY
		6.43.2.5	CHANGE_TO_UID
		6.43.2.6	ErrorLog
		6.43.2.7	ErrorLogEnable
		6.43.2.8	GLOBAL_KILL_SERVER_SWITCH
		6.43.2.9	MAX_CACHE_SIZE_FOR_EACH_FILE_IN_MB
		6.43.2.10	MAX_CACHE_SIZE_IN_MB
		6.43.2.11	MAX_SEPERATE_CACHE_ITEMS
		6.43.2.12	* TemplatesInternalURI
		6.43.2.13	USERNAME_UID_FOR_DAEMON
		6.43.2.14	varSocketTimeoutREAD_seconds
		6.43.2.15	varSocketTimeoutWRITE_seconds
6.44	src/Am	mServerlik	o/server_configuration.h File Reference
	6.44.1	Detailed	Description
	6.44.2	Macro De	efinition Documentation

XXVI

	6.44.2.1	CALCULATE_TIME_FOR_UPLOADS	239
	6.44.2.2	COMPILE_WITH_CLIENT_LIST	239
	6.44.2.3	DEFAULT_SOCKET_READ_TIMEOUT_SECS	240
	6.44.2.4	DEFAULT_SOCKET_WRITE_TIMEOUT_SECS	240
	6.44.2.5	DEFAULT_USERNAME_UID_FOR_DAEMON	240
	6.44.2.6	ENABLE_AUTOMATIC_CONFIGURATION_LOADING	240
	6.44.2.7	ENABLE_COMPRESSION	240
	6.44.2.8	ENABLE_DIRECTORY_LISTING	240
	6.44.2.9	ENABLE_DROPPING_ROOT_UID_IF_ROOT	240
	6.44.2.10	ENABLE_DROPPING_UID_ALWAYS	240
	6.44.2.11	ENABLE_DYNAMIC_CONTENT_COMPRESSION	240
	6.44.2.12	ENABLE_INTERNAL_RESOURCES_RESOLVE	240
	6.44.2.13	ENABLE_POST	240
	6.44.2.14	EPOCH_YEAR_IN_TM_YEAR	241
	6.44.2.15	GROWSTEP_DIRECTORY_LIST_RESPONSE_BODY	241
	6.44.2.16	HTTP_POST_GROWTH_STEP_REQUEST_HEADER	241
	6.44.2.17	INITIAL_DIRECTORY_LIST_RESPONSE_BODY	241
	6.44.2.18	MAX_CLIENT_PRESPAWNED_THREADS	241
	6.44.2.19	MAX_CLIENT_THREADS	241
	6.44.2.20	MAX_CLIENTS_LISTENING_FOR	241
	6.44.2.21	MAX_CLIENTS_PER_IP	241
	6.44.2.22	MAX_CONFIGURATION_FILE_LINE_SIZE	241
	6.44.2.23	MAX_CONTENT_TYPE	241
	6.44.2.24	MAX_DIRECTORY_LIST_RESPONSE_BODY	241
	6.44.2.25	MAX_FILE_READ_BLOCK_KB	242
	6.44.2.26	MAX_HTTP_POST_REQUEST_HEADER	242
	6.44.2.27	MAX_HTTP_REQUEST_HEADER	242
	6.44.2.28	MAX_HTTP_REQUEST_HEADER_LINES	242
	6.44.2.29	MAX_HTTP_REQUEST_HEADER_REPLY	242
	6.44.2.30	MAX_RESOURCE_SLASHES	242
	6.44.2.31	NON_ROOT_UID_IF_USER_FAILS	242
	6.44.2.32	REALLOC_TO_SAVE_MORE_THAN_THIS_NUMBER_BYTES	242
	6.44.2.33	TEMPLATE_INTERNAL_URI	242
	6.44.2.34	THREAD_SLEEP_TIME_FOR_PRESPAWNED_THREADS	242
	6.44.2.35	THREAD_SLEEP_TIME_WHEN_OUR_PRESPAWNED_THREAD_IS_NEXT	243
6.44.3	Function I	Documentation	243
	6.44.3.1	AssignStr	243
	6.44.3.2	EmmitPossibleConfigurationWarnings	243
	6.44.3.3	instance_CountFreeOP	243
	6.44.3.4	instance_CountNewMallocOP	243

CONTENTS xxvii

		6.44.3.5 instance_WeCanCommitMoreMemory
		6.44.3.6 LoadConfigurationFile
		6.44.3.7 SetUsernameAndPassword
	6.44.4	Variable Documentation
		6.44.4.1 AccessLog
		6.44.4.2 AccessLogEnable
		6.44.4.3 CACHING_ENABLED
		6.44.4.4 CHANGE_PRIORITY
		6.44.4.5 CHANGE_TO_UID
		6.44.4.6 ErrorLog
		6.44.4.7 ErrorLogEnable
		6.44.4.8 GLOBAL_KILL_SERVER_SWITCH
		6.44.4.9 MAX_CACHE_SIZE_FOR_EACH_FILE_IN_MB
		6.44.4.10 MAX_CACHE_SIZE_IN_MB
		6.44.4.11 MAX_SEPERATE_CACHE_ITEMS
		6.44.4.12 TemplatesInternalURI
		6.44.4.13 USERNAME_UID_FOR_DAEMON
		6.44.4.14 varSocketTimeoutREAD_seconds
		6.44.4.15 varSocketTimeoutWRITE_seconds
6.45	src/Am	mServerlib/stringscanners/applicationFiles.c File Reference
	6.45.1	Function Documentation
		6.45.1.1 scanFor_applicationFiles
6.46	src/Am	mServerlib/stringscanners/applicationFiles.h File Reference
	6.46.1	Detailed Description
	6.46.2	Enumeration Type Documentation
		6.46.2.1 anonymous enum
	6.46.3	Function Documentation
		6.46.3.1 scanFor_applicationFiles
6.47	src/Am	mServerlib/stringscanners/audioFiles.c File Reference
	6.47.1	Function Documentation
		6.47.1.1 scanFor_audioFiles
6.48	src/Am	mServerlib/stringscanners/audioFiles.h File Reference
	6.48.1	Detailed Description
	6.48.2	Enumeration Type Documentation
		6.48.2.1 anonymous enum
	6.48.3	Function Documentation
		6.48.3.1 scanFor_audioFiles
6.49		mServerlib/stringscanners/firstLines.c File Reference
	6.49.1	Function Documentation
		6.49.1.1 scanFor_firstLines

xxviii CONTENTS

6.50	src/Am	mServerlib/stringscanners/firstLines.h File Reference
	6.50.1	Detailed Description
	6.50.2	Enumeration Type Documentation
		6.50.2.1 anonymous enum
	6.50.3	Function Documentation
		6.50.3.1 scanFor_firstLines
6.51	src/Am	mServerlib/stringscanners/httpHeader.c File Reference
	6.51.1	Function Documentation
		6.51.1.1 scanFor_httpHeader
6.52	src/Am	mServerlib/stringscanners/httpHeader.h File Reference
	6.52.1	Detailed Description
	6.52.2	Enumeration Type Documentation
		6.52.2.1 anonymous enum
	6.52.3	Function Documentation
		6.52.3.1 scanFor_httpHeader
6.53	src/Am	mServerlib/stringscanners/imageFiles.c File Reference
	6.53.1	Function Documentation
		6.53.1.1 scanFor_imageFiles
6.54	src/Am	mServerlib/stringscanners/imageFiles.h File Reference
	6.54.1	Detailed Description
	6.54.2	Enumeration Type Documentation
		6.54.2.1 anonymous enum
	6.54.3	Function Documentation
		6.54.3.1 scanFor_imageFiles
6.55	src/Am	mServerlib/stringscanners/postHeader.c File Reference
	6.55.1	Function Documentation
		6.55.1.1 scanFor_postHeader
6.56	src/Am	mServerlib/stringscanners/postHeader.h File Reference
	6.56.1	Detailed Description
	6.56.2	Enumeration Type Documentation
		6.56.2.1 anonymous enum
	6.56.3	Function Documentation
		6.56.3.1 scanFor_postHeader
6.57	src/Am	mServerlib/stringscanners/textFiles.c File Reference
	6.57.1	Function Documentation
		6.57.1.1 scanFor_textFiles
6.58	src/Am	mServerlib/stringscanners/textFiles.h File Reference
	6.58.1	Detailed Description
	6.58.2	Enumeration Type Documentation
		6.58.2.1 anonymous enum

CONTENTS xxix

	6.58.3	Function Documentation
		6.58.3.1 scanFor_textFiles
6.59	src/Am	mServerlib/stringscanners/videoFiles.c File Reference
	6.59.1	Function Documentation
		6.59.1.1 scanFor_videoFiles
6.60	src/Am	mServerlib/stringscanners/videoFiles.h File Reference
	6.60.1	Detailed Description
	6.60.2	Enumeration Type Documentation
		6.60.2.1 anonymous enum
	6.60.3	Function Documentation
		6.60.3.1 scanFor_videoFiles
6.61	src/Am	mServerlib/threads/freshThreads.c File Reference
	6.61.1	Macro Definition Documentation
		6.61.1.1 MAX_TRIES_TO_FIND_A_THREAD_ID
		6.61.1.2 WEIRD_THING_THAT_WORKS
	6.61.2	Function Documentation
		6.61.2.1 FindAProperThreadID
		6.61.2.2 SpawnThreadToServeNewClient
6.62	src/Am	mServerlib/threads/freshThreads.h File Reference
	6.62.1	Detailed Description
	6.62.2	Function Documentation
		6.62.2.1 SpawnThreadToServeNewClient
6.63	src/Am	mServerlib/threads/prespawnedThreads.c File Reference
	6.63.1	Function Documentation
		6.63.1.1 PreSpawnedThread
		6.63.1.2 PreSpawnThreads
		6.63.1.3 UsePreSpawnedThreadToServeNewClient
6.64	src/Am	mServerlib/threads/prespawnedThreads.h File Reference
	6.64.1	Detailed Description
	6.64.2	Function Documentation
		6.64.2.1 PreSpawnThreads
		6.64.2.2 UsePreSpawnedThreadToServeNewClient
6.65	src/Am	mServerlib/threads/threadedServer.c File Reference
	6.65.1	Function Documentation
		6.65.1.1 HTTPServerlsRunning
		6.65.1.2 MainHTTPServerThread
		6.65.1.3 ServeClient
		6.65.1.4 ServeClientKeepAliveLoop
		6.65.1.5 StartHTTPServer
		6.65.1.6 StopHTTPServer

CONTENTS

6.66	src/Am	mServerlib	o/threads/threadedServer.h File Reference	281
	6.66.1	Detailed I	Description	282
	6.66.2	Function	Documentation	282
		6.66.2.1	HTTPServerIsRunning	282
		6.66.2.2	ServeClient	282
		6.66.2.3	StartHTTPServer	283
		6.66.2.4	StopHTTPServer	284
6.67	src/Am	mServerlib	o/threads/threadInitHelper.c File Reference	285
	6.67.1	Macro De	efinition Documentation	285
		6.67.1.1	SLEEP_FOR_N_NANOSECONDS_WAITING_STACK_MESSAGE	285
	6.67.2	Function	Documentation	285
		6.67.2.1	childFinishedWithParentMessage	285
		6.67.2.2	parentKeepMessageOnStackUntilReady	286
		6.67.2.3	parentKeepMessageOnStackUntilReadyOrTimeout	286
6.68	src/Am	mServerlib	o/threads/threadInitHelper.h File Reference	286
	6.68.1	Detailed I	Description	287
	6.68.2	Function	Documentation	287
		6.68.2.1	childFinishedWithParentMessage	287
		6.68.2.2	parentKeepMessageOnStackUntilReady	288
		6.68.2.3	parentKeepMessageOnStackUntilReadyOrTimeout	288
6.69	src/Am	mServerlib	o/tools/directory_lists.c File Reference	288
	6.69.1	Macro De	efinition Documentation	289
		6.69.1.1	starting	289
		6.69.1.2	tag_after_image	289
		6.69.1.3	tag_pre_image	289
	6.69.2	Function	Documentation	289
		6.69.2.1	GenerateDirectoryPage	289
		6.69.2.2	path_cat	290
6.70	src/Am	mServerlib	o/tools/directory_lists.h File Reference	290
	6.70.1	Detailed I	Description	291
	6.70.2	Function	Documentation	291
		6.70.2.1	GenerateDirectoryPage	291
6.71	src/Am	mServerlib	o/tools/http_tools.c File Reference	292
	6.71.1	Function	Documentation	293
		6.71.1.1	CheckHTTPHeaderCategory	293
		6.71.1.2	CheckHTTPHeaderCategoryAllCaps	294
		6.71.1.3	convertToUpperCase	294
		6.71.1.4	DirectoryExistsAmmServ	294
		6.71.1.5	encodeToBase64	294
		6.71.1.6	FileExistsAmmServ	294

CONTENTS xxxi

	6./1.1./ Filename	StripperOk		 	295
	6.71.1.8 FindInde	File		 	295
	6.71.1.9 findOutC	ientIDOfPeer		 	295
	6.71.1.10 freeString			 	296
	6.71.1.11 GetConte	ntType		 	296
	6.71.1.12 GetConte	ntTypeForExtension		 	297
	6.71.1.13 GetExter	sionImage		 	297
	6.71.1.14 GetExter	tionType		 	297
	6.71.1.15 GetIntFro	mHTTPHeaderFieldPayload		 	298
	6.71.1.16 GetNews	tringFromHTTPHeaderFieldPayl	oad	 	298
	6.71.1.17 ReduceF	athSlashes_Inplace		 	298
	6.71.1.18 Requestl	HTTPWebPage		 	299
	6.71.1.19 seek_bla	nk_char		 	299
	6.71.1.20 seek_no	_blank_char		 	299
	6.71.1.21 ServerTh	reads_DropRootUID		 	299
	6.71.1.22 setSocke	Timeouts		 	299
	6.71.1.23 StripGET	RequestQueryAndFragment		 	300
	6.71.1.24 StripHTM	LCharacters_Inplace		 	300
	6.71.1.25 StripVari	bleFromGETorPOSTString		 	300
	6.71.1.26 stristr			 	301
	6.71.1.27 stristr2Ca	ps		 	301
	6.71.1.28 strToUpc	ase		 	301
	6.71.1.29 trim_last	empty_chars		 	301
6.72 src/Am	nServerlib/tools/http	_tools.h File Reference		 	301
6.72.1	Detailed Descriptio	1		 	303
6.72.2	Typedef Documenta	tion		 	303
	6.72.2.1 contentT	pe		 	303
6.72.3	Enumeration Type	Occumentation		 	303
	6.72.3.1 contentT	peEnumerator		 	303
6.72.4	Function Documen	ation		 	303
	6.72.4.1 CheckHT	TPHeaderCategory		 	303
	6.72.4.2 CheckH1	TPHeaderCategoryAllCaps		 	304
	6.72.4.3 Directory	ExistsAmmServ		 	304
	6.72.4.4 encodeTe	Base64		 	304
	6.72.4.5 FileExists	AmmServ		 	304
	6.72.4.6 Filename	StripperOk		 	305
	6.72.4.7 FindInde	File		 	306
	6.72.4.8 findOutC	ientIDOfPeer		 	306
	6.72.4.9 freeString			 	306
	6.72.4.10 GetConte	ntType		 	307

xxxii CONTENTS

		6.72.4.11	GetDateString	307
		6.72.4.12	GetExtensionImage	307
		6.72.4.13	GetExtentionType	308
		6.72.4.14	GetIntFromHTTPHeaderFieldPayload	308
		6.72.4.15	${\sf GetNewStringFromHTTPHeaderFieldPayload} \ . \ . \ . \ . \ . \ . \ . \ . \ . \ $	309
		6.72.4.16	ReducePathSlashes_Inplace	309
		6.72.4.17	RequestHTTPWebPage	309
		6.72.4.18	seek_blank_char	310
		6.72.4.19	seek_non_blank_char	310
		6.72.4.20	ServerThreads_DropRootUID	310
		6.72.4.21	setSocketTimeouts	310
		6.72.4.22	StripGETRequestQueryAndFragment	310
		6.72.4.23	StripHTMLCharacters_Inplace	310
		6.72.4.24	StripVariableFromGETorPOSTString	311
		6.72.4.25	strToUpcase	311
		6.72.4.26	trim_last_empty_chars	311
6.73	src/Am	mServerlib	/tools/logs.c File Reference	311
	6.73.1	Function	Documentation	312
		6.73.1.1	AccessLogAppend	312
		6.73.1.2	error	312
		6.73.1.3	ErrorLogAppend	312
		6.73.1.4	warning	313
6.74	src/Am	mServerlib	/tools/logs.h File Reference	314
	6.74.1	Detailed [	Description	314
	6.74.2	Macro De	finition Documentation	315
		6.74.2.1	BLACK	315
		6.74.2.2	BLUE	315
		6.74.2.3	BOLDBLACK	315
		6.74.2.4	BOLDBLUE	315
		6.74.2.5	BOLDCYAN	315
		6.74.2.6	BOLDGREEN	315
		6.74.2.7	BOLDMAGENTA	315
		6.74.2.8	BOLDRED	315
		6.74.2.9	BOLDWHITE	315
		6.74.2.10	BOLDYELLOW	315
		6.74.2.11	CYAN	315
		6.74.2.12	GREEN	315
		6.74.2.13	MAGENTA	315
		6.74.2.14	NORMAL	315
		6.74.2.15	RED	315

CONTENTS xxxiii

		6.74.2.16	WHITE
		6.74.2.17	YELLOW
	6.74.3	Function	Documentation
		6.74.3.1	AccessLogAppend
		6.74.3.2	error
		6.74.3.3	ErrorLogAppend
		6.74.3.4	warning
6.75	src/Am	mServerlik	o/tools/time_provider.c File Reference
	6.75.1	Function	Documentation
		6.75.1.1	end_timer
		6.75.1.2	GetDateString
		6.75.1.3	GetTickCountAmmServ
		6.75.1.4	start_timer
	6.75.2	Variable I	Documentation
		6.75.2.1	days
		6.75.2.2	months
6.76	src/Am	mServerlib	o/tools/time_provider.h File Reference
	6.76.1	Detailed	Description
	6.76.2	Function	Documentation
		6.76.2.1	end_timer
		6.76.2.2	GetDateString
		6.76.2.3	GetTickCountAmmServ
		6.76.2.4	start_timer
6.77	src/Am	mServerlib	o/version.h File Reference
	6.77.1	Macro De	efinition Documentation
		6.77.1.1	RC_FILEVERSION
		6.77.1.2	RC_FILEVERSION_STRING
6.78	src/Scr	iptRunner/	main.cpp File Reference
	6.78.1	Macro De	efinition Documentation
		6.78.1.1	ADMIN_BINDING_PORT
		6.78.1.2	DEFAULT_BINDING_PORT
		6.78.1.3	ENABLE_ADMIN_PAGE
		6.78.1.4	ENABLE_CHAT_BOX
		6.78.1.5	ENABLE_PASSWORD_PROTECTION
		6.78.1.6	MAX_BINDING_PORT
		6.78.1.7	MAX_COMMAND_SIZE
	6.78.2	Function	Documentation
		6.78.2.1	close_dynamic_content
		6.78.2.2	execute
		6.78.2.3	getBackCommandLine

CONTENTS

		6.78.2.4	init_dynamic_content	323
		6.78.2.5	joystickExecute	323
		6.78.2.6	main	324
		6.78.2.7	prepare_base_image	324
		6.78.2.8	prepare_form_content_callback	324
		6.78.2.9	prepare_index_content_callback	325
		6.78.2.10	prepare_stats_content_callback	325
		6.78.2.11	prepare_top_image	325
		6.78.2.12	PreplaceChar	325
		6.78.2.13	store_new_configuration_callback	325
		6.78.2.14	termination_handler	325
	6.78.3	Variable [	Documentation	326
		6.78.3.1	admin_root	326
		6.78.3.2	admin_server	326
		6.78.3.3	base_image	326
		6.78.3.4	chatbox	326
		6.78.3.5	default_server	326
		6.78.3.6	form	326
		6.78.3.7	GET_override	326
		6.78.3.8	indexPage	326
		6.78.3.9	page	326
		6.78.3.10	pageLength	326
		6.78.3.11	random_chars	326
		6.78.3.12	settings	326
		6.78.3.13	stats	326
		6.78.3.14	templates_root	326
		6.78.3.15	top_image	326
		6.78.3.16	webserver_root	326
6.79	src/Ser	vices/Scrip	ptRunner/main.cpp File Reference	326
	6.79.1	Macro De	efinition Documentation	328
		6.79.1.1	ADMIN_BINDING_PORT	328
		6.79.1.2	DEFAULT_BINDING_PORT	328
		6.79.1.3	ENABLE_ADMIN_PAGE	328
		6.79.1.4	ENABLE_CHAT_BOX	328
		6.79.1.5	ENABLE_PASSWORD_PROTECTION	328
		6.79.1.6	MAX_BINDING_PORT	328
		6.79.1.7	MAX_COMMAND_SIZE	328
	6.79.2	Function	Documentation	328
		6.79.2.1	close_dynamic_content	328
		6.79.2.2	EraseFile	328

CONTENTS XXXV

		6.79.2.3	execute	328
		6.79.2.4	FileExistsTest	329
		6.79.2.5	getBackCommandLine	329
		6.79.2.6	init_dynamic_content	329
		6.79.2.7	joystickExecute	329
		6.79.2.8	main	330
		6.79.2.9	prepare_base_image	330
		6.79.2.10	prepare_form_content_callback	330
		6.79.2.11	prepare_index_content_callback	331
		6.79.2.12	prepare_stats_content_callback	331
		6.79.2.13	prepare_top_image	331
		6.79.2.14	replaceChar	331
		6.79.2.15	store_new_configuration_callback	331
		6.79.2.16	StringIsHTMLSafe	331
		6.79.2.17	termination_handler	331
	6.79.3	Variable [	Documentation	332
		6.79.3.1	admin_root	332
		6.79.3.2	admin_server	332
		6.79.3.3	base_image	332
		6.79.3.4	chatbox	332
		6.79.3.5	default_server	332
		6.79.3.6	form	332
		6.79.3.7	GET_override	332
		6.79.3.8	indexPage	332
		6.79.3.9	page	332
		6.79.3.10	pageLength	332
		6.79.3.11	random_chars	332
		6.79.3.12	settings	332
		6.79.3.13	stats	332
		6.79.3.14	templates_root	332
		6.79.3.15	top_image	332
		6.79.3.16	webserver_root	332
6.80	src/Stri	ngRecogni	izer/fastStringParser.c File Reference	332
	6.80.1	Macro De	finition Documentation	333
		6.80.1.1	ACTIVATED_LEVELS	333
		6.80.1.2	MAXIMUM_LEVELS	333
			MAXIMUM_LINE_LENGTH	
	6.80.2	Function	Documentation	334
		6.80.2.1	addLevelSpaces	
		6.80.2.2	convertTo_ENUM_ID	334

xxxvi CONTENTS

	6.80.2.3 export_C_Scanner
	6.80.2.4 fastStringParser_addString
	6.80.2.5 fastStringParser_close
	6.80.2.6 fastStringParser_countStringsForNextChar
	6.80.2.7 fastSTringParser_createRulesFromFile
	6.80.2.8 fastStringParser_hasStringsWithNConsecutiveChars
	6.80.2.9 fastStringParser_initialize
	6.80.2.10 printAllEnumeratorItems
	6.80.2.11 printlfAllPossibleStrings
	6.80.2.12 recursiveTraverser
6.8	30.3 Variable Documentation
	6.80.3.1 acceptedChars
	6.80.3.2 fspHTTPHeader
6.81 src	c/StringRecognizer/fastStringParser.h File Reference
6.8	31.1 Detailed Description
6.8	31.2 Function Documentation
	6.81.2.1 export_C_Scanner
	6.81.2.2 fastSTringParser_createRulesFromFile
6.82 src	c/UnitTests/testHashMap.c File Reference
6.8	32.1 Function Documentation
	6.82.1.1 main
Index	340

# **Chapter 1**

# **AmmarServer**

**Author** 

Ammar Qammaz a.k.a. AmmarkoV - http://ammar.gr

A lightweight extendable barebones HTTP server for linux Please see the wiki for more info on whats going on in this repository:) https://github.com/AmmarkoV/AmmarServer/wiki

One of the most basic philosophies behind this is to try to add as much functionality possible in a reusable and very fast way and *WITHOUT* overly increasing lines of code (loc) .. The biggest recent improvements have been actually trying to merge common functionality and reducing loc , and code complexity .

Website: https://github.com/AmmarkoV/RGBDAcquisition

# 1.1 Introduction and History

AmmarServer began as a small sockets project back on 2004, its main use back then was serving as a portable executable that I could take with me to share static files between different machines without having administrator privilages, setting up shares, on different Operating Systems and network topologies..

Needless to say despite beeing "my own brainchild", it wasn't a webserver particularly useful on anything but static content and I always used Apache, MySQL and PHP as infrastructure for serious web-development work which served me well .. until I started working on embedded systems..

The Apache web server is a wonderful piece of software with a very large collection of plugins and modules and a huge percentage of the internet gets served by it every day, it is robust, mature, well documented and it is secure.. But all these positive qualities also mean that it is big and it is complex requiring a relatively large deployment and configuration payload (for a LAMP installation).

Using PHP ( or any other interpreted high-level language ) felt right at home from the first time I used it. With its C-like language structure but more goodies like multi line strings and loose variable declaration rules . It proved to be an invaluable tool but gradually also proved a heavy task for computer hosts lacking many computing resources or serving a very large number of requests. The picture got even worse when services like Wordpress ( which is also great ) that have many thousands lines of code generate dynamic content.. The delays , wether they where Disk , CPU or Memory based summed up and this lead to a very bad user experience while accessing and browsing various site configurations. Of course I am not the only one that has observed this and there are many projects to improve the situation and combat performance overheads such as the Hip-Hop library developed by facebook that translated php to a C++ generating a compiled binary and reduced their loads by a respectable 50%.. Other "home-made" solutions I tried was operating on memfs or ramfs partitions and many other hacks which optimized things more and more..

At some point I thought.. All this is good but is it the best that can be done? What would be the best way to do it? The least overhead possible can only be achieved by closely coupling the webserver with the dynamic content it serves. Compiled php binaries offer a faster way to generate the content but this content is loosely tied with the server that actually sends it. Instead of having seperate "entities" for the webserver the architecture of AmmarServer statically links the webserver library with the dynamic content which is compiled into the same executable.

2 AmmarServer

The simple implementation of AmmarServer

#### 1.2 What Is it?

So the question is, What is it exactly? AmmarServer is a low level framework that allows the creation of binary executables which contain both their webserver and the ability to generate dynamic pages.

A sample application that demonstrates this concept and you can see , is my V4L2ToHTTP project ( https://github.com/AmmarkoV/V4L2ToHTTP ) that uses AmmarServer as its backbone. V4L2ToHTTP is aimed at a thin server that receives frames from a video device ( i.e. webcam ) encodes them into jpeg format in memory , and when a client requests a version of cam.jpg the callback dynamically snaps and uploads a new frame from the camera. The whole point of course is having the minimal possible internal "generation/communication" overhead and the lowest possible memory footprint since the frame is mmaped to the place where the kernel receives the USB camera frame data , it then uses libjpeg for a hardware optimized conversion and then just basically moves a pointer address which is utilized by the send socket command to send the frame. The datapath literally can't get any smaller..

The way to write a web-service using AmmarServer is somewhat different than writing a PHP webservice on Apache. Each service is a different executable (process) that binds a TCP/IP port, instead of a collection of scripts. In order to serve clients each service spawns its own maximum number of threads (and can get individually balanced by the kernel scheduler) instead of preforking seperate processes like Apache does. AmmarServer works with threads (see why) in order to use a lower overall amount of memory, and to make it easier for the programmer (and the Kernel process scheduler) to prioritize serving requests on tight budgets. WebHosting services like godaddy etc.. are not fit to use this model since every "page" would have to be a seperate "server". So this server doesnot target this deployment scenario but rather the "dedicated hosting" one.. It is not really meant for a server that hosts 1000 different PHP services (although I guess it could also be used that way; P) but for a Facebook or Youtube like project when we want a few number of services like uploading, caching and serving content, browsing pictures, editing profile information etc in order to make each of these sub tasks a different "server" as efficient as possible and maintaining a conceptually simple and maintainable model for the developer.

## 1.3 Coding Style

Coding style helpers are kind of a stub for now , since there are key parts of the library that are missing , and providing easier calls , aliases etc for missing functionality is impossible , it is something that should really happen in the future. What I basically want to say is that the Model->View abstractions of Rails , or other modeling techniques are a nice thing , and there isn't any relevant helper functions built in the framework , or even a coding template so unfortunately I can't tell you how to organize your content for now. I would guess that AmmarServer would naturaly mix well with ECPG (PostgreSQL embedded for C ) and that the state information could be kept there if you want to use SQL. Most of the visual things ( CSS , images , videos , audio , JScripts ) should be in static files , and that the callbacks for conent shouldn't be thousands lines of code but instead use external functions that fit the model of the problem you are trying to solve..

There is also extensive testing that has to be done and things related to static string allocation etc .. , things are in a moderate shape right now and can be improved ( for sure )

AmmarServer is relatively stable, but not thoroughly tested (security, pentesting etc) I certainly hope you will find it an interesting and handy codebase.

#### 1.4 Future Plans

Future Planned Projects using AmmarServer

- An opensource RomPager alternative or Webmin alternative
- A more efficient version of myloader (https://github.com/MasterEx/myloader)

1.5 Deployment 3

- · Replacing Apache as the Web Interface of GuarddoG robot
- Making a Video Surveillance daemon like zoneminder with emphasis on performance and small system footprint

Replacing my WebServer (for http://ammar.gr) with a Rasberry pi running AmmarServer

## 1.5 Deployment

To download AmmarServer you can click here (https://github.com/AmmarkoV/AmmarServer/archive/master.-zip) or issue "git clone http://github.com/AmmarkoV/AmmarServer.git" on your terminal To compile it issue mkdir build && cd build && cmake ... && make while beeing in the root directory of the repo.. To run it using the default settings issue ./run\_ammarserver You should review the list of open issues to better inform yourself of the current state of the server

### 1.6 Dependencies

So if you issue sudo apt-get install build-essential (assuming a Debian/Ubuntu based system) you should be able to compile it without problems..

Newer versions also support compression , so you might want to also apt-get install liblzma-dev if you enable ENA-BLE\_COMPRESSION at <a href="mailto:server\_configuration.h">server\_configuration.h</a> MyURL needs libjpeg in order to serve captchas , so to add it sudo apt-get install libjpeg-dev

Compilation is controlled using cmake , so to perform a compilation you just need to issue mkdir build cd build cmake .. make

To update your version of the project you can use the provided script that updates directly from github It will remove any changes you have made to any of the files in the repository ./update\_from\_git from the root directory

AmmarServer

# **Chapter 2**

# **Bug List**

#### Global AMMAR SERVER HTTP HEADER SPEC

A potential bug might arise if the specs of the header file are changed and someone is linking with an older version libAmmServer.a thats why this value exists

# Global AmmServer\_ExecuteCommandLine (char \*command, char \*what2GetBack, unsigned int what2Get-BackMaxSize)

Executing commands can be dangerous, always check and sanitize input before executing, Also be sure about the max size of output so that you don't lose a part of it, also make something like escapeshellcmd

# $\textbf{Global AmmServer\_ReplaceVarInMemoryFile} \ (\textbf{char} * \textbf{page}, \textbf{unsigned int pageLength}, \textbf{char} * \textbf{var}, \textbf{char} * \textbf{value})$

Value should not be bigger than variable otherwise things won't fit in the same memory block

#### Global AmmServer\_SelfCheck (struct AmmServer\_Instance \*instance)

Maybe remove AmmServer SelfCheck

# Global AmmServer\_SignalCountAsBadClientBehaviour (struct AmmServer\_Instance \*instance, struct AmmServer\_DynamicRequest \*rqst)

Client behaviours etc are not implemented yet

#### File AmmServerlib.h

AmmarServer is not properly pentested yet

Global cache\_GetResource (struct AmmServer\_Instance \*instance, struct HTTPHeader \*request, unsigned int resourceCacheID, char \*verified\_filename, unsigned int \*index, unsigned long \*filesize, struct stat \*last\_modification, unsigned char \*compressionSupported, unsigned char \*freeContentAfterUsingIt)

If verified\_filename, is not really verified (i.e. outside of the public\_html root directory, this function could pose a security problem, since it will just blindly open and serve the filename given to it)

**Return values** 

1=Ok,0=Failed

#### File client\_list.h

Client Lists are a stub and not implemented yet

#### File dynamic\_requests.h

Compression should be improved

# File file\_caching.h

File caching relies on hashmap for storing data, so it relies on optimizations done there for seek time optimization, other than that there needs to be a clean-up and code quality improvement

#### File file\_compression.h

Compression should be improved

#### Global GenerateDirectoryPage (char \*system\_path, char \*client\_path, unsigned long \*memoryUsed)

GenerateDirectoryPage does not handle memory correctly, code is in very bad shape

6 Bug List

#### File hashmap.h

This hashmap implementation uses serial searches for now, and needs a lot of work

#### File http header analysis.h

HTTP header analysis can be improved (code style etc.) although the recent use of stringscanners has greatly improved it and reduced lines of code

#### Global LoadConfigurationFile (struct AmmServer Instance \*instance, char \*conf file)

LoadConfigurationFiles etc is not ready yet, although it relies on InputParser and should be easy to implement, there are just things missing still and that's why I postpone implementing it

#### Global MAX CLIENTS PER IP

MAX\_CLIENTS\_PER\_IP is not used if there is no client list declared

#### File post header analysis.h

POST header analysis is not fully implemented yet

#### File prespawnedThreads.h

Prespawned threads have race conditions?

#### File server configuration.h

Server configuration at some point should be ported from defines to a per instance configuration file, some of these defines will always remain since they control global allocations

#### Global StopHTTPServer (struct AmmServer\_Instance \*instance)

Stop web server should be improved, to make sure it unbinds the closing socket

**Return values** 

1=Success,0=Failure

#### **Global TEMPLATE INTERNAL URI**

Please note that the file server has limits for filenames so this should not be very long *asvres*/filename.jpg is OK a filename like *asvres*/filenamemplampla.jpg will return a 404

# **Chapter 3**

# **Data Structure Index**

# 3.1 Data Structures

Here are the data structures with brief descriptions:

AmmServer_DynamicRequest	
When a call to a function that is a dynamic request is done this is the structure that holds the	
information	13
AmmServer_Instance	
This holds all the information about an Ammar Server Instance, sockets, thread pools, cache,	
memory, settings etc, this is the central structure for holding context	14
AmmServer_Instance_Settings	
Each Instance of AmmarServer has some basic settings, which are stored in AmmServer	
Instance_Settings	16
AmmServer_RequestOverride_Context	
We can override/intercept connections before the very fundamental HTTP stage using a request	
override context and AmmServer_AddRequestHandler This is the structure that holds the infor-	
mation and what to be called back to populate the response	17
AmmServer_RH_Context	
We can override resources to respond with our own C function code, to do so a AmmServer	
DynamicRequest must be populated using a AmmServer_AddResourceHandler	18
cache_item	
A cache item and all it's contents	19
clientListContext	
The client list is just a hashmap ( see hashmap.h )	20
fastStringParser	
Internal Structure that holds all the string parser context	21
fspString	
Internal Structure to hold a string and its id for further processing	22
guard_byte	22
hashMap	
The central structure for the hash map	23
hashMapEntry	
An entry on the hash map flattened out for ease of use	24
HTTPHeader	
Each HTTP Request has a header, this is the internal structure that carries the information	
about the header of an HTTP request parsed and ready for easy for consumption by the various	
consumers of HTTP requests	25
HTTPTransaction	
Structure to keep data for an HTTP Transaction	27
Image	28
InputParser	28
InputParserC	33

8 Data Structure Index

PassToHTTPThread PassToHTTPThread	
A structure that holds information to be passed from the main thread to the new (fresh) thread .	34
PassToPreSpawnedThread	36
PreSpawnedThread	
A structure that holds information to be passed from the main thread to the new (prespawned)	
thread	37
time_snap	38
timestamp	
Timestamp for a cache item entry	38
tokens	39
URLDB	40

# Chapter 4

# File Index

# 4.1 File List

Here is a list of all files with br	ief descriptions:
-------------------------------------	-------------------

doc/DoxygenMainpage.h	41
doc/helloworld.c	41
src/AmmCaptcha/AmmCaptcha.h	43
src/AmmCaptcha/imaging.c	98
src/AmmCaptcha/imaging.h	100
· · · · · · · · · · · · · · · · · · ·	101
1 0= 1	102
1 10 1	104
1 10 1	107
src/AmmCaptcha/main.c	46
src/AmmCaptcha/AmmCaptchaTester/main.c	45
src/AmmServerlib.h	
	109
src/AmmServerlib/main.c	53
_ *	231
src/AmmServerlib/server_configuration.h	
	236
	320
	131
src/AmmServerlib/cache/client_list.h	
	135
·	138
src/AmmServerlib/cache/dynamic_requests.h	
	140
	143
src/AmmServerlib/cache/file_caching.h	
Central cache of AmmarServer , it reads/indexes and swaps resources asked by clients for fast	
•	151
src/AmmServerlib/cache/file_compression.c	159
src/AmmServerlib/cache/file_compression.h	
A tool that compresses memory blocks for better bandwidth usage on the expense of computing	
·	162
src/AmmServerlib/hashmap/hashmap.c	164
src/AmmServerlib/hashmap/hashmap.h	
A uniform and clean way to create hashmaps in C and query them	
src/AmmServerlib/header_analysis/http_header_analysis.c	187
src/AmmServerlib/header_analysis/http_header_analysis.h	
Tools to process HTTP requests	192

10 File Index

4.1 File List

src/AmmServerlib/tools/http_tools.h
A collection of tools required by the server and gathered here since they do a very specific job 30
src/AmmServerlib/tools/logs.c
src/AmmServerlib/tools/logs.h
Logging functions
src/AmmServerlib/tools/time_provider.c
src/AmmServerlib/tools/time_provider.h
Timer functions
src/ScriptRunner/main.cpp
src/Services/AmmarServer/main.c
src/Services/GeoPosShare/main.c
src/Services/MyLoader/main.c
src/Services/MyURL/main.c
src/Services/ScriptRunner/main.cpp
src/Services/SimpleTemplate/main.c
src/StringRecognizer/fastStringParser.c
src/StringRecognizer/fastStringParser.h
A tool that converts a file with words ( each word on a new line ) to C code ( see automata ) for
fast string checking
src/StringRecognizer/main.c
src/UnitTests/testHashMap.c

12 File Index

# **Chapter 5**

# **Data Structure Documentation**

# 5.1 AmmServer\_DynamicRequest Struct Reference

When a call to a function that is a dynamic request is done this is the structure that holds the information.

```
#include <AmmServerlib.h>
```

#### **Data Fields**

- · unsigned int headerResponse
- char \* content
- unsigned long contentSize
- unsigned long MAXcontentSize
- char \* compressedContent
- unsigned long compressedContentSize
- unsigned long MAXcompressedContentSize
- · char \* GET\_request
- unsigned int GET\_request\_length
- char \* POST\_request
- unsigned int POST\_request\_length
- unsigned int clientID

# 5.1.1 Detailed Description

When a call to a function that is a dynamic request is done this is the structure that holds the information.

#### 5.1.2 Field Documentation

- 5.1.2.1 unsigned int clientID
- 5.1.2.2 char\* compressedContent
- 5.1.2.3 unsigned long compressedContentSize
- 5.1.2.4 char\* content
- 5.1.2.5 unsigned long contentSize

- 5.1.2.6 char\* GET\_request
- 5.1.2.7 unsigned int GET\_request\_length
- 5.1.2.8 unsigned int headerResponse
- 5.1.2.9 unsigned long MAXcompressedContentSize
- 5.1.2.10 unsigned long MAXcontentSize
- 5.1.2.11 char\* POST\_request
- 5.1.2.12 unsigned int POST\_request\_length

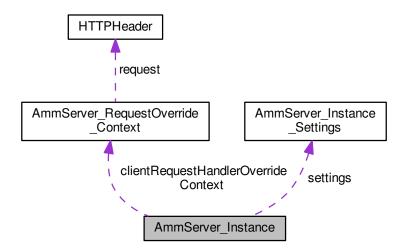
• src/AmmServerlib/AmmServerlib.h

# 5.2 AmmServer\_Instance Struct Reference

This holds all the information about an Ammar Server Instance , sockets , thread pools , cache , memory , settings etc , this is the central structure for holding context.

```
#include <AmmServerlib.h>
```

Collaboration diagram for AmmServer\_Instance:



#### **Data Fields**

- char instanceName [MAX\_INSTANCE\_NAME\_STRING]
- struct AmmServer Instance Settings settings
- · unsigned int prespawn turn to serve
- · unsigned int prespawn\_jobs\_started

- · unsigned int prespawn\_jobs\_finished
- int files\_open
- · int serversock
- · int server running
- int pause\_server
- int stop\_server
- unsigned long loaded\_cache\_items\_Kbytes
- unsigned int loaded\_cache\_items
- void \* cache
- void \* cacheHashMap
- void \* clientList
- unsigned int CLIENT\_THREADS\_STARTED
- unsigned int CLIENT\_THREADS\_STOPPED
- pthread\_t server\_thread\_id
- pthread\_t \* threads\_pool
- void \* prespawned\_pool
- struct

AmmServer\_RequestOverride\_Context \* clientRequestHandlerOverrideContext

- · char webserver root [MAX FILE PATH]
- char templates\_root [MAX\_FILE\_PATH]

#### 5.2.1 Detailed Description

This holds all the information about an Ammar Server Instance , sockets , thread pools , cache , memory , settings etc , this is the central structure for holding context.

#### 5.2.2 Field Documentation

- 5.2.2.1 void\* cache
- 5.2.2.2 void\* cacheHashMap
- 5.2.2.3 unsigned int CLIENT\_THREADS\_STARTED
- 5.2.2.4 unsigned int CLIENT\_THREADS\_STOPPED
- 5.2.2.5 void\* clientList
- 5.2.2.6 struct AmmServer\_RequestOverride\_Context\* clientRequestHandlerOverrideContext
- 5.2.2.7 int files\_open
- 5.2.2.8 char instanceName[MAX\_INSTANCE\_NAME\_STRING]
- 5.2.2.9 unsigned int loaded\_cache\_items
- 5.2.2.10 unsigned long loaded\_cache\_items\_Kbytes
- 5.2.2.11 int pause\_server
- 5.2.2.12 unsigned int prespawn\_jobs\_finished
- 5.2.2.13 unsigned int prespawn\_jobs\_started

```
5.2.2.14 unsigned int prespawn_turn_to_serve

5.2.2.15 void* prespawned_pool

5.2.2.16 int server_running

5.2.2.17 pthread_t server_thread_id

5.2.2.18 int serversock

5.2.2.19 struct AmmServer_Instance_Settings settings

5.2.2.20 int stop_server

5.2.2.21 char templates_root[MAX_FILE_PATH]

5.2.2.22 pthread_t* threads_pool
```

• src/AmmServerlib/AmmServerlib.h

5.2.2.23 char webserver\_root[MAX FILE PATH]

# 5.3 AmmServer\_Instance\_Settings Struct Reference

Each Instance of AmmarServer has some basic settings , which are stored in AmmServer\_Instance\_Settings.

```
#include <AmmServerlib.h>
```

#### **Data Fields**

- int PASSWORD PROTECTION
- char \* USERNAME
- char \* PASSWORD
- char \* BASE64PASSWORD
- int BINDING\_PORT

#### 5.3.1 Detailed Description

Each Instance of AmmarServer has some basic settings, which are stored in AmmServer\_Instance\_Settings.

- 5.3.2 Field Documentation
- 5.3.2.1 char\* BASE64PASSWORD
- 5.3.2.2 int BINDING\_PORT
- 5.3.2.3 char\* PASSWORD
- 5.3.2.4 int PASSWORD\_PROTECTION

#### 5.3.2.5 char\* USERNAME

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

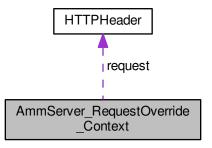
• src/AmmServerlib/AmmServerlib.h

# 5.4 AmmServer\_RequestOverride\_Context Struct Reference

We can override/intercept connections before the very fundamental HTTP stage using a request override context and AmmServer\_AddRequestHandler This is the structure that holds the information and what to be called back to populate the response.

```
#include <AmmServerlib.h>
```

Collaboration diagram for AmmServer\_RequestOverride\_Context:



#### **Data Fields**

- char requestHeader [64]
- struct HTTPHeader \* request
- void \* request\_override\_callback

#### 5.4.1 Detailed Description

We can override/intercept connections before the very fundamental HTTP stage using a request override context and AmmServer\_AddRequestHandler This is the structure that holds the information and what to be called back to populate the response.

#### 5.4.2 Field Documentation

- 5.4.2.1 struct HTTPHeader\* request
- 5.4.2.2 void\* request\_override\_callback
- 5.4.2.3 char requestHeader[64]

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

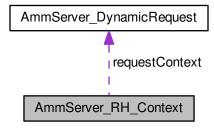
• src/AmmServerlib/AmmServerlib.h

# 5.5 AmmServer RH Context Struct Reference

We can override resources to respond with our own C function code , to do so a AmmServer\_DynamicRequest must be populated using a AmmServer\_AddResourceHandler.

#include <AmmServerlib.h>

Collaboration diagram for AmmServer\_RH\_Context:



#### **Data Fields**

- unsigned int RH\_Scenario
- · unsigned int last callback
- unsigned int callback\_every\_x\_msec
- char callback\_cooldown
- void \* dynamicRequestCallbackFunction
- char web\_root\_path [MAX\_FILE\_PATH]
- char resource\_name [MAX\_RESOURCE]
- struct AmmServer\_DynamicRequest requestContext

#### 5.5.1 Detailed Description

We can override resources to respond with our own C function code , to do so a AmmServer\_DynamicRequest must be populated using a AmmServer\_AddResourceHandler.

#### 5.5.2 Field Documentation

- 5.5.2.1 char callback\_cooldown
- 5.5.2.2 unsigned int callback\_every\_x\_msec
- 5.5.2.3 void\* dynamicRequestCallbackFunction
- 5.5.2.4 unsigned int last\_callback
- 5.5.2.5 struct AmmServer\_DynamicRequest requestContext

- 5.5.2.6 char resource\_name[MAX\_RESOURCE]
- 5.5.2.7 unsigned int RH\_Scenario
- 5.5.2.8 char web\_root\_path[MAX\_FILE\_PATH]

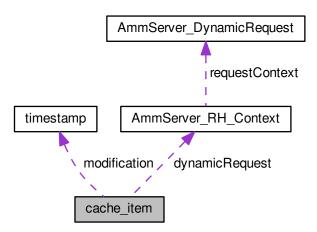
• src/AmmServerlib/AmmServerlib.h

# 5.6 cache\_item Struct Reference

A cache item and all it's contents.

#include <file\_caching.h>

Collaboration diagram for cache\_item:



#### **Data Fields**

- void \* dynamicRequestCallbackFunction
- struct AmmServer\_RH\_Context \* dynamicRequest
- unsigned char doNOTCacheRule
- char \* content
- unsigned long \* contentSize
- char \* compressedContent
- unsigned long \* compressedContentSize
- contentType contentTypeID
- · struct timestamp modification

#### 5.6.1 Detailed Description

A cache item and all it's contents.

#### 5.6.2 Field Documentation

- 5.6.2.1 char\* compressedContent
- 5.6.2.2 unsigned long\* compressedContentSize
- 5.6.2.3 char\* content
- 5.6.2.4 unsigned long\* contentSize
- 5.6.2.5 contentType contentTypeID
- 5.6.2.6 unsigned char doNOTCacheRule
- 5.6.2.7 struct AmmServer\_RH\_Context\* dynamicRequest
- 5.6.2.8 void\* dynamicRequestCallbackFunction
- 5.6.2.9 struct timestamp modification

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

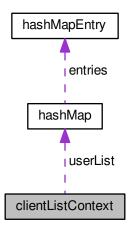
· src/AmmServerlib/cache/file caching.h

#### 5.7 clientListContext Struct Reference

The client list is just a hashmap ( see hashmap.h )

```
#include <client_list.h>
```

Collaboration diagram for clientListContext:



#### **Data Fields**

struct hashMap \* userList

#### 5.7.1 Detailed Description

The client list is just a hashmap ( see hashmap.h )

#### 5.7.2 Field Documentation

#### 5.7.2.1 struct hashMap\* userList

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

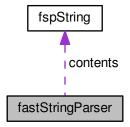
• src/AmmServerlib/cache/client list.h

# 5.8 fastStringParser Struct Reference

Internal Structure that holds all the string parser context.

#include <fastStringParser.h>

Collaboration diagram for fastStringParser:



#### **Data Fields**

- struct fspString \* contents
- unsigned int stringsLoaded
- unsigned int MAXstringsLoaded
- char \* functionName
- · unsigned int shortestStringLength
- unsigned int longestStringLength

# 5.8.1 Detailed Description

Internal Structure that holds all the string parser context.

#### 5.8.2 Field Documentation

#### 5.8.2.1 struct fspString\* contents

- 5.8.2.2 char\* functionName
- 5.8.2.3 unsigned int longestStringLength
- 5.8.2.4 unsigned int MAXstringsLoaded
- 5.8.2.5 unsigned int shortestStringLength
- 5.8.2.6 unsigned int stringsLoaded

• src/StringRecognizer/fastStringParser.h

# 5.9 fspString Struct Reference

Internal Structure to hold a string and its id for further processing.

```
#include <fastStringParser.h>
```

#### **Data Fields**

- char \* str
- char \* strIDFriendly
- · unsigned int strLength

#### 5.9.1 Detailed Description

Internal Structure to hold a string and its id for further processing.

#### 5.9.2 Field Documentation

- 5.9.2.1 char\* str
- 5.9.2.2 char\* strIDFriendly
- 5.9.2.3 unsigned int strLength

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

• src/StringRecognizer/fastStringParser.h

## 5.10 guard\_byte Struct Reference

```
#include <InputParser_C.h>
```

#### **Data Fields**

· unsigned int checksum

#### 5.10.1 Field Documentation

#### 5.10.1.1 unsigned int checksum

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

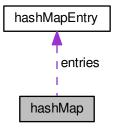
• src/AmmServerlib/InputParser/InputParser\_C.h

# 5.11 hashMap Struct Reference

The central structure for the hash map.

```
#include <hashmap.h>
```

Collaboration diagram for hashMap:



#### **Data Fields**

- unsigned int maxNumberOfEntries
- unsigned int curNumberOfEntries
- unsigned int entryAllocationStep
- struct hashMapEntry \* entries
- void \* clearItemCallbackFunction
- pthread\_mutex\_t hm\_addLock
- pthread\_mutex\_t hm\_fileLock

## 5.11.1 Detailed Description

The central structure for the hash map.

## 5.11.2 Field Documentation

- 5.11.2.1 void\* clearItemCallbackFunction
- 5.11.2.2 unsigned int curNumberOfEntries
- 5.11.2.3 struct hashMapEntry\* entries

- 5.11.2.4 unsigned int entryAllocationStep
- 5.11.2.5 pthread\_mutex\_t hm\_addLock
- 5.11.2.6 pthread\_mutex\_t hm\_fileLock
- 5.11.2.7 unsigned int maxNumberOfEntries

• src/AmmServerlib/hashmap/hashmap.h

# 5.12 hashMapEntry Struct Reference

An entry on the hash map flattened out for ease of use.

```
#include <hashmap.h>
```

#### **Data Fields**

- · unsigned long keyHash
- unsigned int keyLength
- char \* key
- · unsigned int payloadLength
- void \* payload
- · unsigned int hits

### 5.12.1 Detailed Description

An entry on the hash map flattened out for ease of use.

#### 5.12.2 Field Documentation

- 5.12.2.1 unsigned int hits
- 5.12.2.2 char\* key
- 5.12.2.3 unsigned long keyHash
- 5.12.2.4 unsigned int keyLength
- 5.12.2.5 void\* payload
- 5.12.2.6 unsigned int payloadLength

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

src/AmmServerlib/hashmap/hashmap.h

#### 5.13 HTTPHeader Struct Reference

Each HTTP Request has a header, this is the internal structure that carries the information about the header of an HTTP request parsed and ready for easy for consumption by the various consumers of HTTP requests.

#include <AmmServerlib.h>

#### **Data Fields**

- char \* headerRAW
- · unsigned int headerRAWSize
- int requestType
- char resource [MAX\_RESOURCE+1]
- char verified\_local\_resource [MAX\_FILE\_PATH+1]
- char GETquery [MAX QUERY+1]
- char \* POSTrequest
- unsigned long POSTrequestSize
- · unsigned char authorized
- unsigned char keepalive
- unsigned char supports\_compression
- · unsigned long range\_start
- unsigned long range\_end
- unsigned long ContentLength
- char \* cookie
- · unsigned int cookieLength
- · char \* host
- unsigned int hostLength
- char \* referer
- · unsigned int refererLength
- char \* eTag
- unsigned int eTagLength
- char \* userAgent
- · unsigned int userAgentLength
- char \* contentType
- unsigned int contentTypeLength
- char \* contentDisposition
- unsigned int contentDispositionLength
- char \* boundary
- · unsigned int boundaryLength

#### 5.13.1 Detailed Description

Each HTTP Request has a header, this is the internal structure that carries the information about the header of an HTTP request parsed and ready for easy for consumption by the various consumers of HTTP requests.

#### 5.13.2 Field Documentation

- 5.13.2.1 unsigned char authorized
- 5.13.2.2 char\* boundary
- 5.13.2.3 unsigned int boundaryLength

5.13.2.4	char* contentDisposition
5.13.2.5	unsigned int contentDispositionLength
5.13.2.6	unsigned long ContentLength
5.13.2.7	char* contentType
5.13.2.8	unsigned int contentTypeLength
5.13.2.9	char* cookie
5.13.2.10	unsigned int cookieLength
5.13.2.11	char∗ eTag
5.13.2.12	unsigned int eTagLength
5.13.2.13	char GETquery[MAX_QUERY+1]
5.13.2.14	char∗ headerRAW
5.13.2.15	unsigned int headerRAWSize
5.13.2.16	char∗ host
5.13.2.17	unsigned int hostLength
5.13.2.18	unsigned char keepalive
5.13.2.19	char∗ POSTrequest
5.13.2.20	unsigned long POSTrequestSize
5.13.2.21	unsigned long range_end
5.13.2.22	unsigned long range_start
5.13.2.23	char∗ referer
5.13.2.24	unsigned int refererLength
5.13.2.25	int requestType
5.13.2.26	char resource[MAX_RESOURCE+1]
5.13.2.27	unsigned char supports_compression
5.13.2.28	char∗ userAgent
5.13.2.29	unsigned int userAgentLength
5.13.2.30	char verified_local_resource[MAX_FILE_PATH+1]

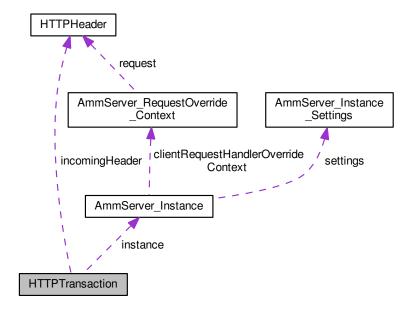
• src/AmmServerlib/AmmServerlib.h

#### 5.14 HTTPTransaction Struct Reference

Structure to keep data for an HTTP Transaction.

#include <AmmServerlib.h>

Collaboration diagram for HTTPTransaction:



#### **Data Fields**

- struct AmmServer\_Instance \* instance
- struct HTTPHeader incomingHeader
- char \* outgoingBody
- unsigned int outgoingBodySize
- unsigned int resourceCacheID
- int clientSock
- unsigned int clientListID
- · unsigned int threadID
- int prespawnedThreadFlag

### 5.14.1 Detailed Description

Structure to keep data for an HTTP Transaction.

#### 5.14.2 Field Documentation

5.14.2.1 unsigned int clientListID

5.14.2.2 int clientSock

```
5.14.2.3 struct HTTPHeader incomingHeader
5.14.2.4 struct AmmServer_Instance* instance
5.14.2.5 char* outgoingBody
5.14.2.6 unsigned int outgoingBodySize
5.14.2.7 int prespawnedThreadFlag
5.14.2.8 unsigned int resourceCacheID
```

5.14.2.9 unsigned int threadID

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

• src/AmmServerlib/AmmServerlib.h

# 5.15 Image Struct Reference

```
#include <imaging.h>
```

#### **Data Fields**

- unsigned char \* pixels
- · unsigned int width
- · unsigned int height
- · unsigned int depth
- · unsigned int imageSize

#### 5.15.1 Field Documentation

- 5.15.1.1 unsigned int depth
- 5.15.1.2 unsigned int height
- 5.15.1.3 unsigned int imageSize
- 5.15.1.4 unsigned char\* pixels
- 5.15.1.5 unsigned int width

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

• src/AmmCaptcha/imaging.h

## 5.16 InputParser Class Reference

#include <InputParser.h>

#### **Public Member Functions**

- const char \* Version ()
- void DefaultDelimeterSetup ()
- InputParser ()
- ∼InputParser ()
- void SetDelimeter (int num, char tmp)
- char GetDelimeter (int num)
- unsigned int GetWord (int num, char \*thestr, unsigned int thestrsize)
- unsigned int GetUpcaseWord (int num, char \*thestr, unsigned int thestrsize)
- unsigned int GetLowercaseWord (int num, char \*thestr, unsigned int thestrsize)
- char GetWordChar (int num, int chr)
- signed int GetWordInt (int num)
- unsigned short GetWordLength (int num)
- int SeperateWords (char \*inpt)
- int SeperateWordsCC (const char \*inpt)
- int SeperateWordsUC (unsigned char \*inpt)

#### 5.16.1 Constructor & Destructor Documentation

#### 5.16.1.1 InputParser()

Here is the call graph for this function:



#### 5.16.1.2 ∼InputParser ( )

Here is the call graph for this function:



### 5.16.2 Member Function Documentation

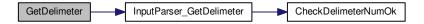
#### 5.16.2.1 void DefaultDelimeterSetup ( )

Here is the call graph for this function:



#### 5.16.2.2 char GetDelimeter (int num)

Here is the call graph for this function:



#### 5.16.2.3 unsigned int GetLowercaseWord (int num, char \* thestr, unsigned int thestrsize)

Here is the call graph for this function:



#### 5.16.2.4 unsigned int GetUpcaseWord ( int num, char \* thestr, unsigned int thestrsize )

Here is the call graph for this function:



#### 5.16.2.5 unsigned int GetWord ( int num, char \* thestr, unsigned int thestrsize )

Here is the call graph for this function:



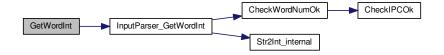
#### 5.16.2.6 char GetWordChar ( int num, int chr )

Here is the call graph for this function:



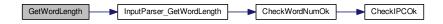
#### 5.16.2.7 signed int GetWordInt (int num)

Here is the call graph for this function:



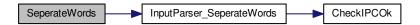
#### 5.16.2.8 unsigned short GetWordLength (int num)

Here is the call graph for this function:



#### 5.16.2.9 int SeperateWords ( char \* inpt )

Here is the call graph for this function:



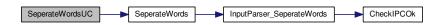
#### 5.16.2.10 int SeperateWordsCC ( const char \* inpt )

Here is the call graph for this function:



#### 5.16.2.11 int SeperateWordsUC ( unsigned char \* inpt )

Here is the call graph for this function:



# 5.16.2.12 void SetDelimeter ( int num, char tmp )

Here is the call graph for this function:



#### 5.16.2.13 const char\* Version ( )

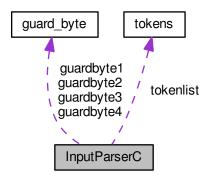
The documentation for this class was generated from the following files:

- src/AmmServerlib/InputParser/InputParser.h
- src/AmmServerlib/InputParser/InputParser.cpp

## 5.17 InputParserC Struct Reference

#include <InputParser\_C.h>

Collaboration diagram for InputParserC:



#### **Data Fields**

- struct guard\_byte guardbyte1
- unsigned int str\_length
- unsigned char local\_allocation
- char \* str
- struct guard\_byte guardbyte2
- unsigned short cur\_container\_count
- unsigned short max\_container\_count
- char \* container\_start
- char \* container\_end
- unsigned short cur\_delimeter\_count
- unsigned short max\_delimeter\_count
- char \* delimeters
- struct guard\_byte guardbyte3
- unsigned int tokens\_max
- unsigned int tokens\_count
- struct tokens \* tokenlist
- struct guard\_byte guardbyte4

#### 5.17.1 Field Documentation

- 5.17.1.1 char\* container\_end
- 5.17.1.2 char\* container\_start
- 5.17.1.3 unsigned short cur\_container\_count
- 5.17.1.4 unsigned short cur\_delimeter\_count

5.17.1.5	char* delimeters
5.17.1.6	struct guard_byte guardbyte1
5.17.1.7	struct guard_byte guardbyte2
5.17.1.8	struct guard_byte guardbyte3
5.17.1.9	struct guard_byte guardbyte4
5.17.1.10	unsigned char local_allocation
5.17.1.11	unsigned short max_container_count
5.17.1.12	unsigned short max_delimeter_count
5.17.1.13	char* str
5.17.1.14	unsigned int str_length
5.17.1.15	struct tokens* tokenlist
5.17.1.16	unsigned int tokens_count
5.17.1.17	unsigned int tokens_max
<b>T</b>	and the first the state of the

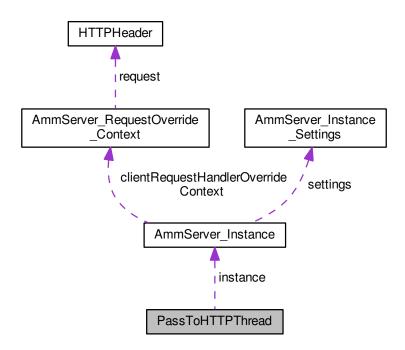
• src/AmmServerlib/InputParser/InputParser\_C.h

# 5.18 PassToHTTPThread Struct Reference

A structure that holds information to be passed from the main thread to the new (fresh) thread.

#include <freshThreads.h>

Collaboration diagram for PassToHTTPThread:



#### **Data Fields**

- volatile int keep\_var\_on\_stack
- struct sockaddr\_in client
- unsigned int clientlen
- unsigned int thread\_id
- · unsigned int port
- · int clientsock
- struct AmmServer\_Instance \* instance
- int pre\_spawned\_thread

#### 5.18.1 Detailed Description

A structure that holds information to be passed from the main thread to the new (fresh) thread.

#### 5.18.2 Field Documentation

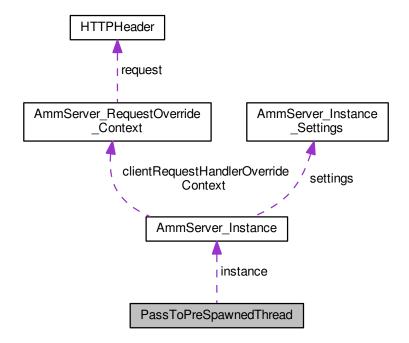
- 5.18.2.1 struct sockaddr\_in client
- 5.18.2.2 unsigned int clientlen
- 5.18.2.3 int clientsock
- 5.18.2.4 struct AmmServer\_Instance\* instance

- 5.18.2.5 volatile int keep\_var\_on\_stack
- 5.18.2.6 unsigned int port
- 5.18.2.7 int pre\_spawned\_thread
- 5.18.2.8 unsigned int thread\_id

• src/AmmServerlib/threads/freshThreads.h

# 5.19 PassToPreSpawnedThread Struct Reference

Collaboration diagram for PassToPreSpawnedThread:



#### **Data Fields**

- struct AmmServer\_Instance \* instance
- unsigned int i\_adapt

#### 5.19.1 Field Documentation

#### 5.19.1.1 unsigned int i\_adapt

#### 5.19.1.2 struct AmmServer\_Instance\* instance

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

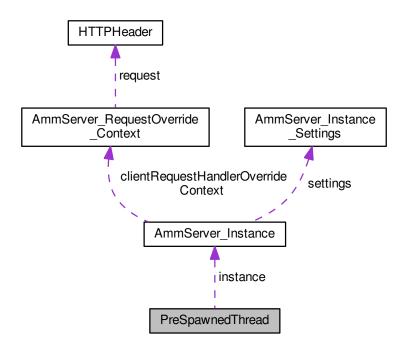
• src/AmmServerlib/threads/prespawnedThreads.c

### 5.20 PreSpawnedThread Struct Reference

A structure that holds information to be passed from the main thread to the new (prespawned) thread.

#include prespawnedThreads.h>

Collaboration diagram for PreSpawnedThread:



### **Data Fields**

- · volatile unsigned char busy
- unsigned int threadNum
- struct AmmServer\_Instance \* instance
- pthread\_t thread\_id
- · int clientsock
- · struct sockaddr in client
- · unsigned int clientlen
- char webserver\_root [MAX\_FILE\_PATH]
- char templates\_root [MAX\_FILE\_PATH]

### 5.20.1 Detailed Description

A structure that holds information to be passed from the main thread to the new (prespawned) thread.

#### 5.20.2 Field Documentation

- 5.20.2.1 volatile unsigned char busy
- 5.20.2.2 struct sockaddr\_in client
- 5.20.2.3 unsigned int clientlen
- 5.20.2.4 int clientsock
- 5.20.2.5 struct AmmServer Instance\* instance
- 5.20.2.6 char templates\_root[MAX\_FILE\_PATH]
- 5.20.2.7 pthread\_t thread\_id
- 5.20.2.8 unsigned int threadNum
- 5.20.2.9 char webserver\_root[MAX\_FILE\_PATH]

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

• src/AmmServerlib/threads/prespawnedThreads.h

### 5.21 time\_snap Struct Reference

```
#include <time_provider.h>
```

#### **Data Fields**

· struct timeval starttime endtime difference

### 5.21.1 Field Documentation

5.21.1.1 struct timeval starttime endtime difference

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

• src/AmmServerlib/tools/time\_provider.h

### 5.22 timestamp Struct Reference

Timestamp for a cache item entry.

#include <file\_caching.h>

#### **Data Fields**

- unsigned char hour
- unsigned char minute
- · unsigned char second
- · unsigned char wday
- · unsigned char day
- unsigned char month
- · unsigned int year

### 5.22.1 Detailed Description

Timestamp for a cache item entry.

#### 5.22.2 Field Documentation

- 5.22.2.1 unsigned char day
- 5.22.2.2 unsigned char hour
- 5.22.2.3 unsigned char minute
- 5.22.2.4 unsigned char month
- 5.22.2.5 unsigned char second
- 5.22.2.6 unsigned char wday
- 5.22.2.7 unsigned int year

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

• src/AmmServerlib/cache/file\_caching.h

### 5.23 tokens Struct Reference

```
#include <InputParser_C.h>
```

### **Data Fields**

- · unsigned int token start
- · unsigned int length

### 5.23.1 Field Documentation

- 5.23.1.1 unsigned int length
- 5.23.1.2 unsigned int token\_start

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

src/AmmServerlib/InputParser/InputParser\_C.h

### 5.24 URLDB Struct Reference

### **Data Fields**

- char \* longURL
- char \* shortURL
- unsigned long shortURLHash

### 5.24.1 Field Documentation

- 5.24.1.1 char\* longURL
- 5.24.1.2 char\* shortURL
- 5.24.1.3 unsigned long shortURLHash

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

• src/Services/MyURL/main.c

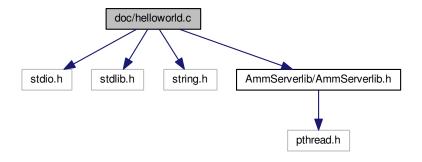
# **Chapter 6**

## **File Documentation**

### 6.1 doc/DoxygenMainpage.h File Reference

### 6.2 doc/helloworld.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "AmmServerlib/AmmServerlib.h"
Include dependency graph for helloworld.c:
```



#### **Functions**

- void \* prepare\_helloworld\_content\_callback (unsigned int associated\_vars)
- void init\_dynamic\_content ()
- void close\_dynamic\_content ()
- int main (int argc, char \*argv[])

#### **Variables**

- char webserver\_root [512] ="public\_html/"
- char templates\_root [512] ="public\_html/templates/"
- struct AmmServer\_RH\_Context helloworld ={0}

• unsigned int helloworld\_times\_shown =0

### 6.2.1 Function Documentation

### 6.2.1.1 void close\_dynamic\_content ( )

Here is the call graph for this function:



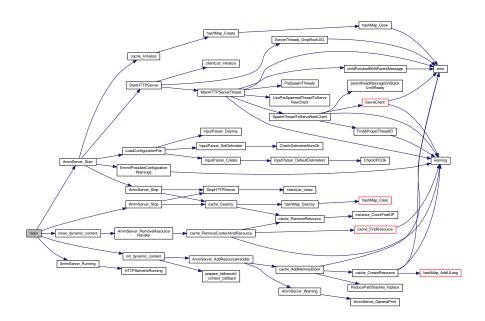
### 6.2.1.2 void init\_dynamic\_content ( )

Here is the call graph for this function:



### 6.2.1.3 int main ( int argc, char \* argv[] )

Dynamic content code ..! END -----



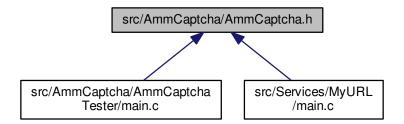
- 6.2.1.4 void\* prepare\_helloworld\_content\_callback ( unsigned int associated\_vars )
- 6.2.2 Variable Documentation
- 6.2.2.1 struct AmmServer RH Context helloworld ={0}

Dynamic content code ..! START!

- 6.2.2.2 unsigned int helloworld\_times\_shown =0
- 6.2.2.3 char templates\_root[512] ="public\_html/templates/"
- 6.2.2.4 char webserver\_root[512] ="public\_html/"

### 6.3 src/AmmCaptcha/AmmCaptcha.h File Reference

This graph shows which files directly or indirectly include this file:

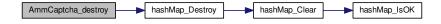


#### **Functions**

- int AmmCaptcha\_initialize (char \*font, char \*dictFilename)
- int AmmCaptcha\_destroy ()
- int AmmCaptcha\_isReplyCorrect (unsigned int captchalD, char \*reply)
- int AmmCaptcha\_getCaptchaFrame (unsigned int captchaID, char \*mem, unsigned long \*mem\_size)
- int testAmmCaptcha ()

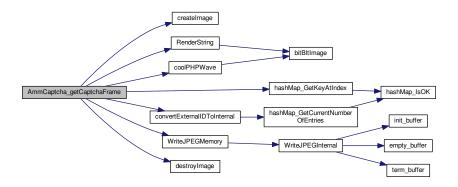
#### 6.3.1 Function Documentation

6.3.1.1 int AmmCaptcha\_destroy ( )



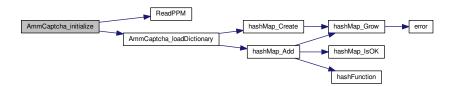
#### 6.3.1.2 int AmmCaptcha\_getCaptchaFrame ( unsigned int captchalD, char \* mem, unsigned long \* mem\_size )

Here is the call graph for this function:

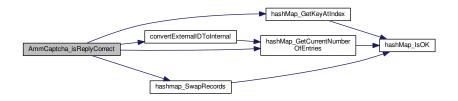


### 6.3.1.3 int AmmCaptcha\_initialize ( char \* font, char \* dictFilename )

Here is the call graph for this function:

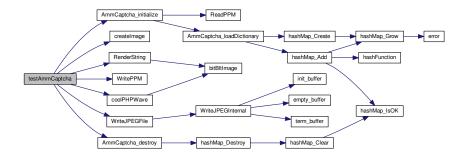


### 6.3.1.4 int AmmCaptcha\_isReplyCorrect ( unsigned int captchalD, char \* reply )



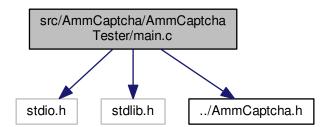
#### 6.3.1.5 int testAmmCaptcha ( )

Here is the call graph for this function:



### 6.4 src/AmmCaptcha/AmmCaptchaTester/main.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include "../AmmCaptcha.h"
Include dependency graph for main.c:
```



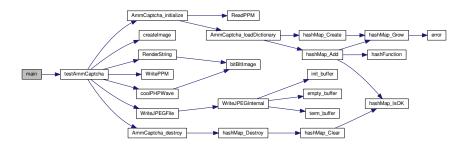
### **Functions**

• int main ()

### 6.4.1 Function Documentation

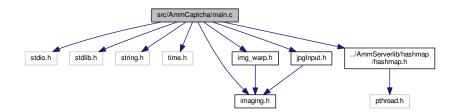
```
6.4.1.1 int main ( )
```

Here is the call graph for this function:



### 6.5 src/AmmCaptcha/main.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include "imaging.h"
#include "img_warp.h"
#include "jpgInput.h"
#include "../AmmServerlib/hashmap/hashmap.h"
Include dependency graph for main.c:
```



### Macros

• #define RANDOMIZE AFTER FAILED ATTEMPT 1

### **Functions**

- int RenderString (struct Image \*frame, struct Image \*font, unsigned int x, unsigned int y, char \*str)
- unsigned int convertExternalIDToInternal (unsigned int captchalD)
- int AmmCaptcha\_isReplyCorrect (unsigned int captchalD, char \*reply)
- int AmmCaptcha\_getCaptchaFrame (unsigned int captchalD, char \*mem, unsigned long \*mem\_size)
- int AmmCaptcha\_copyCaptchaJPEGImageWithCopy (unsigned int captchaID, char \*mem, unsigned long \*mem\_size)
- int AmmCaptcha\_loadDictionary (char \*dictFilename)
- int AmmCaptcha\_initialize (char \*font, char \*dictFilename)

- int AmmCaptcha\_destroy ()
- int testAmmCaptcha ()

#### **Variables**

- unsigned int fontX = 19
- unsigned int fontY = 22
- struct Image fontRAW ={0}
- struct hashMap \* captchaStrings =0

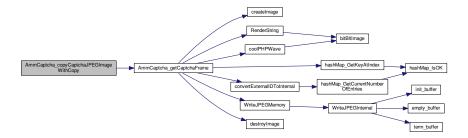
### 6.5.1 Macro Definition Documentation

6.5.1.1 #define RANDOMIZE\_AFTER\_FAILED\_ATTEMPT 1

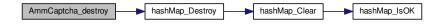
### 6.5.2 Function Documentation

6.5.2.1 int AmmCaptcha\_copyCaptchaJPEGImageWithCopy ( unsigned int *captchalD*, char \* *mem*, unsigned long \* *mem\_size* )

Here is the call graph for this function:

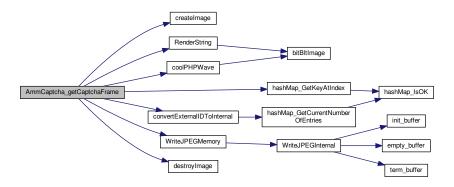


### 6.5.2.2 int AmmCaptcha\_destroy ( )



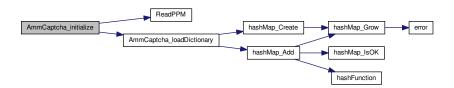
6.5.2.3 int AmmCaptcha\_getCaptchaFrame ( unsigned int captchalD, char \* mem, unsigned long \* mem\_size )

Here is the call graph for this function:

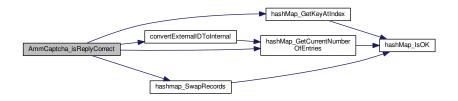


6.5.2.4 int AmmCaptcha\_initialize ( char \* font, char \* dictFilename )

Here is the call graph for this function:

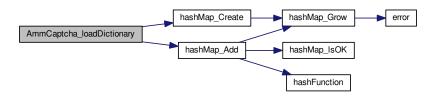


6.5.2.5 int AmmCaptcha\_isReplyCorrect ( unsigned int captchalD, char \* reply )



### 6.5.2.6 int AmmCaptcha\_loadDictionary ( char \* dictFilename )

Here is the call graph for this function:



#### 6.5.2.7 unsigned int convertExternalIDToInternal ( unsigned int captchalD )

Here is the call graph for this function:

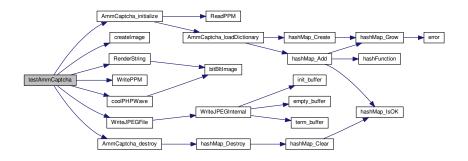


### 6.5.2.8 int RenderString ( struct Image \* frame, struct Image \* font, unsigned int x, unsigned int y, char \* str )



#### 6.5.2.9 int testAmmCaptcha ( )

Here is the call graph for this function:



#### 6.5.3 Variable Documentation

- 6.5.3.1 struct hashMap\* captchaStrings =0
- 6.5.3.2 struct Image fontRAW ={0}
- 6.5.3.3 unsigned int fontX = 19
- 6.5.3.4 unsigned int fontY = 22

### 6.6 src/AmmServerlib/InputParser/InputParser\_C\_Tester/main.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include "../InputParser_C.h"
#include <time.h>
Include dependency graph for main.c:
```

src/AmmServerlib/InputParser/InputParser\_C\_Tester/main.c

../InputParser\_C.h time.h

stdio.h stdlib.h string.h ctype.h

### **Macros**

• #define max\_ret\_word 256

### **Functions**

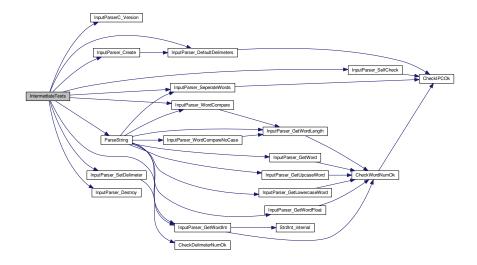
- void ParseString (struct InputParserC \*ipc, char \*thestr)
- int IntermediateTests ()
- int main ()

#### 6.6.1 Macro Definition Documentation

6.6.1.1 #define max\_ret\_word 256

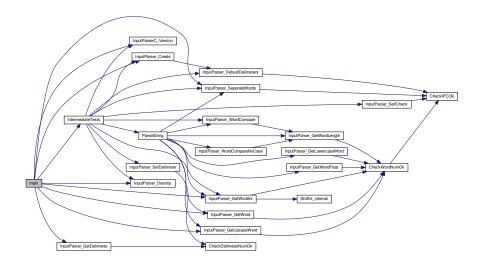
### 6.6.2 Function Documentation

6.6.2.1 int IntermediateTests ( )

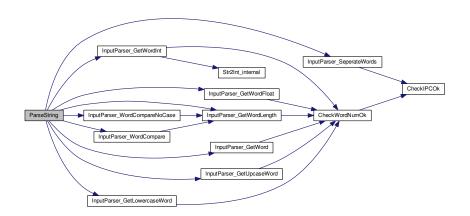


### 6.6.2.2 int main ( )

Here is the call graph for this function:



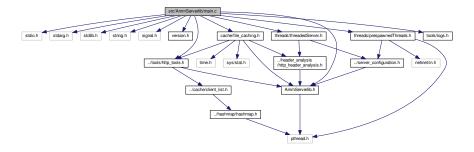
### 6.6.2.3 void ParseString ( struct InputParserC \* ipc, char \* thestr )



#### 6.7 src/AmmServerlib/main.c File Reference

```
#include <stdio.h>
#include <stdarg.h>
#include <stdlib.h>
#include <string.h>
#include <signal.h>
#include "version.h"
#include "AmmServerlib.h"
#include "threads/threadedServer.h"
#include "threads/prespawnedThreads.h"
#include "cache/file_caching.h"
#include "tools/http_tools.h"
#include "tools/logs.h"
```

Include dependency graph for main.c:



#### **Functions**

char \* AmmServer\_Version ()

Returns a string with the version of AmmarServer, in case it returns NULL it means that we are linked to AmmarServerNULL which means a fake binary.

• int AmmServer\_CheckIfHeaderBinaryAreTheSame (int headerSpec)

Internal Check to compare against changes of the header files.

- void AmmServer\_GeneralPrint (char \*color, char \*label, const char \*format, va\_list \*arglist)
- void AmmServer\_Warning (const char \*format,...)

Writes the C string pointed by format to stderr, as a warning (Yellow) and logs it to the appropriate log If format includes format specifiers (subsequences beginning with %), the additional arguments following format are formatted and inserted in the resulting string replacing their respective specifiers.

void AmmServer Error (const char \*format,...)

Writes the C string pointed by format to stderr, as an error (Red) and logs it to the appropriate log If format includes format specifiers (subsequences beginning with %), the additional arguments following format are formatted and inserted in the resulting string replacing their respective specifiers.

• void AmmServer Success (const char \*format,...)

Writes the C string pointed by format to stderr, as a success ( Green ) and logs it to the appropriate log If format includes format specifiers (subsequences beginning with %), the additional arguments following format are formatted and inserted in the resulting string replacing their respective specifiers.

• int AmmServer Stop (struct AmmServer Instance \*instance)

Stop a Web Server, deallocate memory, free ports and free the server instance..

• struct AmmServer\_Instance \* AmmServer\_Start (char \*name, char \*ip, unsigned int port, char \*conf\_file, char \*web\_root\_path, char \*templates\_root\_path)

Start a Web Server, allocate memory, bind ports and return its instance..

• struct AmmServer\_Instance \* AmmServer\_StartWithArgs (char \*name, int argc, char \*\*argv, char \*ip, unsigned int port, char \*conf\_file, char \*web\_root\_path, char \*templates\_root\_path)

Start a Web Server, allocate memory, bind ports and return its instance, also process arguments (argc and argv from int main(int argc, char \*argv[])).

int AmmServer\_Running (struct AmmServer\_Instance \*instance)

Query if an instance of AmmarServer is initialized and running.

 int AmmServer\_AddRequestHandler (struct AmmServer\_Instance \*instance, struct AmmServer\_Request-Override Context \*RequestOverrideContext, char \*request type, void \*callback)

Add a request handler to handle requests, before they get processed internally Calling this will bind a C function that will be called and produce output when someone asks for any resource using the specified method TODO: Improve this documenatation.

• int AmmServer\_AddResourceHandler (struct AmmServer\_Instance \*instance, struct AmmServer\_RH\_-Context \*context, char \*resource\_name, char \*web\_root, unsigned int allocate\_mem\_bytes, unsigned int callback every x msec, void \*callback, unsigned int scenario)

Add a request handler to handle dynamic requests, the core mechanic of AmmarServer Calling this will bind a C function that will be called and produce output when someone asks for a resource TODO: Improve this documenatation.

- int AmmServer\_PreCacheFile (struct AmmServer\_Instance \*instance, char \*filename)
- int AmmServer\_DoNOTCacheResourceHandler (struct AmmServer\_Instance \*instance, struct AmmServer\_RH Context \*context)

Set resource handler to no-cache mode, this means whoever asks for it will never get a cached response.

- int AmmServer\_DoNOTCacheResource (struct AmmServer\_Instance \*instance, char \*resource\_name)
  - Set resource to no-cache mode, this means whoever asks for it will never get a cached response.
- int AmmServer\_RemoveResourceHandler (struct AmmServer\_Instance \*instance, struct AmmServer\_RH\_-Context \*context, unsigned char free mem)

Remove a request handler that hanles dynamic requests.

int AmmServer\_GetInfo (struct AmmServer\_Instance \*instance, unsigned int info\_type)

Get an Integer out of the state of an instance , of course one can dive into the instance structure but this is a much more clean way to do this.

• int AmmServer\_POSTArg (struct AmmServer\_Instance \*instance, struct AmmServer\_DynamicRequest \*rqst, char \*var\_id\_IN, char \*var\_value\_OUT, unsigned int max\_var\_value\_OUT)

Get a POST argument.

• int AmmServer\_GETArg (struct AmmServer\_Instance \*instance, struct AmmServer\_DynamicRequest \*rqst, char \*var\_id\_IN, char \*var\_value\_OUT, unsigned int max\_var\_value\_OUT)

Get a GET argument.

• int AmmServer\_FILES (struct AmmServer\_Instance \*instance, struct AmmServer\_DynamicRequest \*rqst, char \*var\_id\_IN, char \*var\_value\_OUT, unsigned int max\_var\_value\_OUT)

Access a FILE submitted by a dynamic requested.

• int \_POST (struct AmmServer\_Instance \*instance, struct AmmServer\_DynamicRequest \*rqst, char \*var\_id-\_IN, char \*var\_value\_OUT, unsigned int max\_var\_value\_OUT)

Shorthand/Shortcut for AmmServer\_POSTArg()

• int \_GET (struct AmmServer\_Instance \*instance, struct AmmServer\_DynamicRequest \*rqst, char \*var\_id\_l-N, char \*var\_value\_OUT, unsigned int max\_var\_value\_OUT)

Shorthand/Shortcut for AmmServer\_GETArg()

• int \_FILES (struct AmmServer\_Instance \*instance, struct AmmServer\_DynamicRequest \*rqst, char \*var\_id\_IN, char \*var\_value\_OUT, unsigned int max\_var\_value\_OUT)

Shorthand/Shortcut for AmmServer FILES()

int AmmServer\_SignalCountAsBadClientBehaviour (struct AmmServer\_Instance \*instance, struct AmmServer\_DynamicRequest \*rqst)

Staged way to easily handle bad clients etc from the clients, currently a stub..!

int AmmServer\_SaveDynamicRequest (char \*filename, struct AmmServer\_Instance \*instance, struct AmmServer\_DynamicRequest \*rqst)

Save Dynamic Request to file.

• int AmmServer GetIntSettingValue (struct AmmServer Instance \*instance, unsigned int set type)

Get an Integer out of the state of an instance , of course one can dive into the instance structure but this is a much more clean way to do this.

int AmmServer\_SetIntSettingValue (struct AmmServer\_Instance \*instance, unsigned int set\_type, int set\_value)

Set an Integer inside the state of an instance, of course one can dive into the instance structure but this is a much more clean way to do this.

char \* AmmServer\_GetStrSettingValue (struct AmmServer\_Instance \*instance, unsigned int set\_type)

Get a String out of the state of an instance, of course one can dive into the instance structure but this is a much more clean way to do this.

• int AmmServer\_SetStrSettingValue (struct AmmServer\_Instance \*instance, unsigned int set\_type, char \*set\_value)

Set an string inside the state of an instance , of course one can dive into the instance structure but this is a much more clean way to do this.

• struct AmmServer\_Instance \* AmmServer\_StartAdminInstance (char \*ip, unsigned int port)

Planned functionality for a default http administrator panel per server per instance, currently not implemented correctly.

int AmmServer\_SelfCheck (struct AmmServer\_Instance \*instance)

Perform a sanity check on the instance of AmmarServer, this is mostly a dev debug tool and an entry point for code inside AmmServerlib.

• int AmmServer ReplaceVarInMemoryFile (char \*page, unsigned int pageLength, char \*var, char \*value)

Hot-Replace a variable inside a memory block, typically used to replace placeholders inside text files, like \$\$\$\$\$\$\$NAME\$\$\$\$\$\$\$, the value should be smaller or equal to the var beeing replaced.

- void AmmServer GlobalTerminationHandler (int signum)
- int AmmServer RegisterTerminationSignal (void \*callback)

Register a function to call a function that gracefully terminates a client when a SIGKILL or the time to stop the server comes.

 int AmmServer\_ExecuteCommandLine (char \*command, char \*what2GetBack, unsigned int what2GetBack-MaxSize)

Execute a command and copy its output to the provided buffer.

char \* AmmServer\_ReadFileToMemory (char \*filename, unsigned int \*length)

Read a file and store it to a freshly allocated memory block.

• int AmmServer\_WriteFileFromMemory (char \*filename, char \*memory, unsigned int memoryLength)

Dump a memory block to a file.

int AmmServer\_DirectoryExists (char \*filename)

Check if directory Exists.

• int AmmServer FileExists (char \*filename)

Check if file Exists.

• int AmmServer\_EraseFile (char \*filename)

Frase a File

unsigned int AmmServer\_StringIsHTMLSafe (char \*str)

Check if a string has html elements inside it, so if we append it to a web site we won't have html injected.

#### **Variables**

void(\* TerminationCallback )()=0

#### 6.7.1 Function Documentation

6.7.1.1 int \_FILES ( struct AmmServer\_Instance \* instance, struct AmmServer\_DynamicRequest \* rqst, char \* var\_id\_IN, char \* var\_value\_OUT, unsigned int max\_var\_value\_OUT )

Shorthand/Shortcut for AmmServer FILES()

Here is the call graph for this function:



6.7.1.2 int \_GET ( struct AmmServer\_Instance \* instance, struct AmmServer\_DynamicRequest \* rqst, char \* var\_id\_IN, char \* var\_value\_OUT, unsigned int max\_var\_value\_OUT)

Shorthand/Shortcut for AmmServer GETArg()

Here is the call graph for this function:



6.7.1.3 int \_POST ( struct AmmServer\_Instance \* instance, struct AmmServer\_DynamicRequest \* rqst, char \* var\_id\_IN, char \* var\_value\_OUT, unsigned int max\_var\_value\_OUT )

Shorthand/Shortcut for AmmServer\_POSTArg()

Here is the call graph for this function:



6.7.1.4 int AmmServer\_AddRequestHandler ( struct AmmServer\_Instance \* instance, struct AmmServer\_RequestOverride\_Context \* RequestOverrideContext, char \* request\_type, void \* callback )

Add a request handler to handle requests , before they get processed internally Calling this will bind a C function that will be called and produce output when someone asks for any resource using the specified method TODO : Improve this documenatation.

Parameters

An	AmmarServer Instance

Α	AmmServer_RequestOverride_Context to be populated
Request	Туре
Pointer	to function callback

#### **Return values**

1=Success,0=Fail	

Here is the call graph for this function:



6.7.1.5 int AmmServer\_AddResourceHandler ( struct AmmServer\_Instance \* instance, struct

AmmServer\_RH\_Context \* context, char \* resource\_name, char \* web\_root, unsigned int allocate\_mem\_bytes,
unsigned int callback\_every\_x\_msec, void \* callback, unsigned int scenario )

Add a request handler to handle dynamic requests , the core mechanic of AmmarServer Calling this will bind a C function that will be called and produce output when someone asks for a resource TODO : Improve this documenatation.

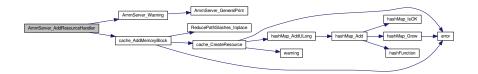
#### **Parameters**

An	AmmarServer Instance
An	AmmServer_RH_Context to be populated
Name	of resource that should get dynamic responses (i.e. "index.html")
Root	Path for the specific resource
Memory	chunk to allocate for responses , ( this is the max response size )
Minimum	time between two calls of the function (0 = no minimum time)
Function	to be called and provides output when someone asks for resource
Scenario/Profile	of this resource ( see RHScenarios )

#### Return values

1=Success,0=Fail	

Here is the call graph for this function:



6.7.1.6 int AmmServer\_CheckIfHeaderBinaryAreTheSame ( int headerSpec )

Internal Check to compare against changes of the header files.

#### **Parameters**

Header	be AMMAR_SERVER_HTTP_HEADER_SPEC
--------	----------------------------------

#### Return values

```
1=Success,0=Failure
```

6.7.1.7 int AmmServer\_DirectoryExists ( char \* filename )

Check if directory Exists.

#### **Parameters**

Path	to directory
------	--------------

#### Return values

1=Exists,0=Does	not Exist

Here is the call graph for this function:



6.7.1.8 int AmmServer\_DoNOTCacheResource ( struct AmmServer\_Instance \* instance, char \* resource\_name )

Set resource to no-cache mode, this means whoever asks for it will never get a cached response.

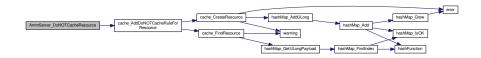
#### **Parameters**

Instance	of an AmmarServer
Resource	name that we want to always serve fresh

#### Return values

```
1=Success,0=Failure
```

Here is the call graph for this function:



6.7.1.9 int AmmServer\_DoNOTCacheResourceHandler ( struct AmmServer\_Instance \* instance, struct AmmServer\_RH\_Context \* context )

Set resource handler to no-cache mode, this means whoever asks for it will never get a cached response.

#### **Parameters**

Instance	of an AmmarServer
Resource	context that should always be served fresh ( AmmServer_RH_Context )

#### Return values

```
1=Success,0=Failure
```

Here is the call graph for this function:



6.7.1.10 int AmmServer\_EraseFile ( char \* filename )

Erase a File.

#### **Parameters**

Path   to file	Path	to file
----------------	------	---------

#### Return values

```
1=Success,0=Failure
```

6.7.1.11 void AmmServer\_Error ( const char \* format, ... )

Writes the C string pointed by format to stderr , as an error ( Red ) and logs it to the appropriate log If format includes format specifiers (subsequences beginning with %), the additional arguments following format are formatted and inserted in the resulting string replacing their respective specifiers.

#### **Parameters**

format,see	<pre>printf(http://www.cplusplus.com/reference/cstdio/printf/)</pre>
Arbitrary	number of other parameters that where defined in format

Here is the call graph for this function:



6.7.1.12 int AmmServer\_ExecuteCommandLine ( char \* command, char \* what2GetBack, unsigned int what2GetBackMaxSize )

Execute a command and copy its output to the provided buffer.

#### **Parameters**

Command	to execute
Allocated	memory to store the result
Size	of Allocated memory

#### Return values

1=Ok,0=Failed	

**Bug** Executing commands can be dangerous, always check and sanitize input before executing, Also be sure about the max size of output so that you don't lose a part of it, also make something like escapeshellcmd

6.7.1.13 int AmmServer\_FileExists ( char \* filename )

Check if file Exists.

#### **Parameters**

Path	to file

#### Return values

1=Exists,0=Does	not Exist
-----------------	-----------

Here is the call graph for this function:



6.7.1.14 int AmmServer\_FILES ( struct AmmServer\_Instance \* instance, struct AmmServer\_DynamicRequest \* rqst, char \* var\_id\_IN, char \* var\_value\_OUT, unsigned int max\_var\_value\_OUT)

Access a FILE submitted by a dynamic requested.

#### **Parameters**

Instance	of an AmmarServer
Request	that contains the POST argument ( see AmmServer_DynamicRequest )
Input	Name of argument we are looking for
Output	Pointer that will be copied with the value we were looking for
Maximum	Size for output Value

### Return values

1=Success,0=Failure	

6.7.1.15 void AmmServer\_GeneralPrint ( char \* color, char \* label, const char \* format, va\_list \* arglist )

6.7.1.16 int AmmServer\_GETArg ( struct AmmServer\_Instance \* instance, struct AmmServer\_DynamicRequest \* rqst, char \* var\_id\_IN, char \* var\_value\_OUT, unsigned int max\_var\_value\_OUT)

Get a GET argument.

#### **Parameters**

Instance	of an AmmarServer
Request	that contains the POST argument ( see AmmServer_DynamicRequest )
Input	Name of argument we are looking for
Output	Pointer that will be copied with the value we were looking for
Maximum	Size for output Value

#### Return values

1=Success,0=Failure	

Here is the call graph for this function:



### 6.7.1.17 int AmmServer\_GetInfo ( struct AmmServer\_Instance \* instance, unsigned int info\_type )

Get an Integer out of the state of an instance , of course one can dive into the instance structure but this is a much more clean way to do this.

#### **Parameters**

An	AmmarServer Instance
An	ID about which info we want , see ( AmmServInfos )

### Return values

Value	of the integer we asked about

### 6.7.1.18 int AmmServer\_GetIntSettingValue ( struct AmmServer\_Instance \* instance, unsigned int set\_type )

Get an Integer out of the state of an instance , of course one can dive into the instance structure but this is a much more clean way to do this.

#### **Parameters**

An	AmmarServer Instance
An	ID about which integer info we want , see ( AmmServSettings )

#### Return values

Value	of the integer we asked about
-------	-------------------------------

### 6.7.1.19 char\* AmmServer\_GetStrSettingValue ( struct AmmServer\_Instance \* instance, unsigned int set\_type )

Get a String out of the state of an instance , of course one can dive into the instance structure but this is a much more clean way to do this.

#### **Parameters**

An	AmmarServer Instance
An	ID about which string info we want , see ( AmmServStrSettings )

#### Return values

Value	of the string we asked about

- 6.7.1.20 void AmmServer\_GlobalTerminationHandler ( int signum )
- 6.7.1.21 int AmmServer\_POSTArg ( struct AmmServer\_Instance \* instance, struct AmmServer\_DynamicRequest \* rqst, char \* var\_id\_IN, char \* var\_value\_OUT, unsigned int max\_var\_value\_OUT)

Get a POST argument.

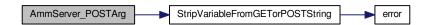
#### **Parameters**

Instance	of an AmmarServer
Request	that contains the POST argument ( see AmmServer_DynamicRequest )
Input	Name of argument we are looking for
Output	Pointer that will be copied with the value we were looking for
Maximum	Size for output Value

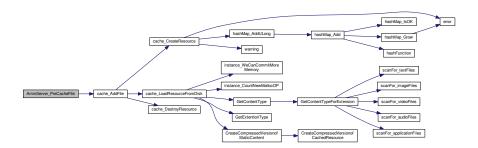
#### Return values

1=Success,0=Failure	

Here is the call graph for this function:



 $\textbf{6.7.1.22} \quad \text{int } \textbf{AmmServer\_PreCacheFile} \, ( \, \, \textbf{struct } \, \textbf{AmmServer\_Instance} \, * \, \textit{instance}, \, \, \textbf{char} \, * \, \textit{filename} \, \, )$ 



6.7.1.23 char\* AmmServer\_ReadFileToMemory ( char\* filename, unsigned int\* length)

Read a file and store it to a freshly allocated memory block.

#### **Parameters**

	Input	Filename
Г	Output	Maximum Size

#### Return values

Pointer	to the new memory or 0=Failed
	··· · · · · · · · · · · · · · · · · ·

Here is the call graph for this function:



### 6.7.1.24 int AmmServer\_RegisterTerminationSignal ( void \* callback )

Register a function to call a function that gracefully terminates a client when a SIGKILL or the time to stop the server comes.

#### **Parameters**

D	1
Pointer	I TO TUNCTION
1 Ollitoi	to idiotion

#### Return values

1=Exists,0=Does	not Exist

Here is the call graph for this function:



# 6.7.1.25 int AmmServer\_RemoveResourceHandler ( struct AmmServer\_Instance \* instance, struct AmmServer\_RH\_Context \* context, unsigned char free\_mem )

Remove a request handler that hanles dynamic requests.

#### **Parameters**

An	AmmarServer Instance
An	AmmServer_RH_Context to be freed
Switch	to control freeing memory or not for this context (typically should be set to 1 except one
	knows what he is trying to do )

#### **Return values**

```
1=Success,0=Failure
```

Here is the call graph for this function:



6.7.1.26 int AmmServer\_ReplaceVarInMemoryFile ( char \* page, unsigned int pageLength, char \* var, char \* value )

Hot-Replace a variable inside a memory block , typically used to replace placeholders inside text files , like \$\$\$\$\$\$NAME\$\$\$\$\$\$, the value should be smaller or equal to the var beeing replaced.

#### **Parameters**

Pointer	to memory that contains the document
Size	of document
Variable	to be replaced
What	to replace it with

#### **Return values**

1=Ok.0=Failed	
1-Onjo-r and	

Bug Value should not be bigger than variable otherwise things won't fit in the same memory block

6.7.1.27 int AmmServer\_Running ( struct AmmServer\_Instance \* instance )

Query if an instance of AmmarServer is initialized and running.

#### **Parameters**

An	AmmarServer Instance

### Return values

```
1=Running,0=Stopped
```



6.7.1.28 int AmmServer\_SaveDynamicRequest ( char \* filename, struct AmmServer\_Instance \* instance, struct AmmServer\_DynamicRequest \* rqst )

Save Dynamic Request to file.

#### **Parameters**

Filename	to save the dynamic request
Instance	of an AmmarServer
Request	that we want to save to a file ( see AmmServer_DynamicRequest )

#### Return values

1=Success,0=Failure	

Here is the call graph for this function:



6.7.1.29 int AmmServer\_SelfCheck ( struct AmmServer\_Instance \* instance )

Perform a sanity check on the instance of AmmarServer, this is mostly a dev debug tool and an entry point for code inside AmmServerlib.

#### **Parameters**

Ammar	Server Instance
Return values	

#### TIOLUTTI VUIGOO

1=Ok,0=Failed	

Bug Maybe remove AmmServer\_SelfCheck

6.7.1.30 int AmmServer\_SetIntSettingValue ( struct AmmServer\_Instance \* instance, unsigned int set\_type, int set\_value )

Set an Integer inside the state of an instance , of course one can dive into the instance structure but this is a much more clean way to do this.

#### **Parameters**

An	AmmarServer Instance
An	ID about which integer info we want , see ( AmmServSettings )
New	value to set

#### Return values

Value	of the integer we asked about
-------	-------------------------------

6.7.1.31 int AmmServer\_SetStrSettingValue ( struct AmmServer\_Instance \* instance, unsigned int set\_type, char \* set\_value )

Set an string inside the state of an instance , of course one can dive into the instance structure but this is a much more clean way to do this.

#### **Parameters**

An	AmmarServer Instance
An	ID about which integer info we want , see ( AmmServStrSettings )
New	string value to set

#### Return values

1=Success,0=Failure	

Here is the call graph for this function:



6.7.1.32 int AmmServer\_SignalCountAsBadClientBehaviour ( struct AmmServer\_Instance \* instance, struct AmmServer\_DynamicRequest \* rqst )

Staged way to easily handle bad clients etc from the clients, currently a stub..!

Bug Client behaviours etc are not implemented yet

6.7.1.33 struct AmmServer\_Instance\* AmmServer\_Start ( char \* name, char \* ip, unsigned int port, char \* conf\_file, char \* web\_root\_path, char \* templates\_root\_path )

Start a Web Server, allocate memory, bind ports and return its instance..

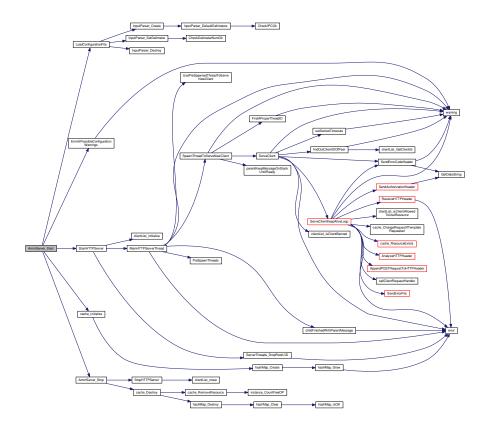
#### **Parameters**

String	containing the name of this Server
String	containing the IP to be binded ( 0.0.0.0 , for all interfaces )
Port	to use , ports under 1000 require superuser privileges
String	with the filename of a configuration file
String	with the root public_html directory , all directories that are childs of this dir could be visible
String	with the root directory for templates ( custom 404 pages etc )

#### Return values

An A	Ammar Server instance or 0=Failure
------	------------------------------------

Here is the call graph for this function:



6.7.1.34 struct AmmServer\_Instance\* AmmServer\_StartAdminInstance ( char \* ip, unsigned int port )

Planned functionality for a default http administrator panel per server per instance, currently not implemented correctly.

### **Parameters**

IP	to bind the interface at
Port	to use

#### Return values

Value	of the integer we asked about

6.7.1.35 struct AmmServer\_Instance\* AmmServer\_StartWithArgs ( char \* name, int argc, char \*\* argv, char \* ip, unsigned int port, char \* conf\_file, char \* web\_root\_path, char \* templates\_root\_path )

Start a Web Server , allocate memory , bind ports and return its instance , also process arguments ( argc and argv from int main(int argc, char \*argv[]) ) ..

#### **Parameters**

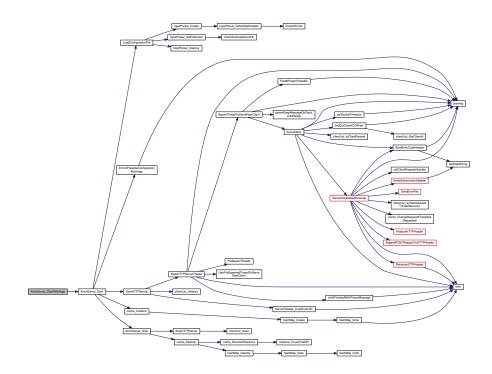
String	containing the name of this Server

argc,number	of arguments
argv,array	of strings
String	containing the IP to be binded (0.0.0.0, for all interfaces)
Port	to use , ports under 1000 require superuser privileges
String	with the filename of a configuration file
String	with the root public_html directory , all directories that are childs of this dir could be visible
String	with the root directory for templates ( custom 404 pages etc )

#### Return values

|--|

Here is the call graph for this function:



### 6.7.1.36 int AmmServer\_Stop ( struct AmmServer\_Instance \* instance )

Stop a Web Server , deallocate memory , free ports and free the server instance..

### **Parameters**

An	AmmarServer Instance
----	----------------------

#### Return values

1=Success,0=Fallure	1_Suggest 0_Eailura
---------------------	---------------------

Here is the call graph for this function:



#### 6.7.1.37 unsigned int AmmServer\_StringlsHTMLSafe ( char \* str )

Check if a string has html elements inside it, so if we append it to a web site we won't have html injected.

#### **Parameters**

Input	String
Return values	

```
6.7.1.38 void AmmServer_Success ( const char * format, ... )
```

1=Safe,0=Unsafe

Writes the C string pointed by format to stderr, as a success ( Green ) and logs it to the appropriate log If format includes format specifiers (subsequences beginning with %), the additional arguments following format are formatted and inserted in the resulting string replacing their respective specifiers.

#### **Parameters**

format,see	<pre>printf(http://www.cplusplus.com/reference/cstdio/printf/)</pre>
Arbitrary	number of other parameters that where defined in format

Here is the call graph for this function:



6.7.1.39 char\* AmmServer\_Version ( )

Returns a string with the version of AmmarServer , in case it returns NULL it means that we are linked to AmmarServerNULL which means a fake binary.

6.7.1.40 void AmmServer\_Warning ( const char \* format, ... )

Writes the C string pointed by format to stderr, as a warning (Yellow) and logs it to the appropriate log If format includes format specifiers (subsequences beginning with %), the additional arguments following format are formatted

and inserted in the resulting string replacing their respective specifiers.

#### **Parameters**

format,see	<pre>printf(http://www.cplusplus.com/reference/cstdio/printf/)</pre>
Arbitrary	number of other parameters that where defined in format

Here is the call graph for this function:



6.7.1.41 int AmmServer\_WriteFileFromMemory ( char \* filename, char \* memory, unsigned int memoryLength )

Dump a memory block to a file.

#### **Parameters**

Output	Filename
Input	Pointer to memory
Size	of memory block

#### Return values

```
1=Ok,0=Failed
```

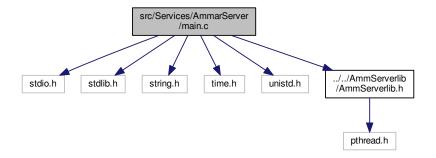
## 6.7.2 Variable Documentation

6.7.2.1 void( \* TerminationCallback)()=0

# 6.8 src/Services/AmmarServer/main.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <unistd.h>
#include "../../AmmServerlib/AmmServerlib.h"
```

Include dependency graph for main.c:



#### **Macros**

- #define MAX\_BINDING\_PORT 65534
- #define ENABLE PASSWORD PROTECTION 0
- #define ENABLE\_CHAT\_BOX 0
- #define DEFAULT\_BINDING\_PORT 8080
- #define ADMIN BINDING PORT 8082
- #define ENABLE ADMIN PAGE 0
- #define MAX SCRIPT RESPONSE SIZE 40960

### **Functions**

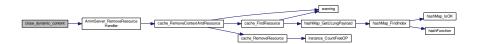
- void \* prepare\_chatbox\_content\_callback (struct AmmServer\_DynamicRequest \*rqst)
- void \* prepare\_stats\_content\_callback (struct AmmServer\_DynamicRequest \*rqst)
- void \* prepare\_random\_content\_callback (struct AmmServer\_DynamicRequest \*rqst)
- void \* prepare\_form\_content\_callback (struct AmmServer\_DynamicRequest \*rqst)
- void \* prepare\_gps\_content\_callback (struct AmmServer\_DynamicRequest \*rqst)
- void \* executeScriptFunction (struct AmmServer\_DynamicRequest \*rqst)
- void \* request\_override\_callback (char \*content)
- void init dynamic content ()
- void close\_dynamic\_content ()
- int main (int argc, char \*argv[])

#### **Variables**

- char admin root [MAX FILE PATH] = "admin html/"
- char webserver root [MAX\_FILE\_PATH] ="public\_html/"
- char templates\_root [MAX\_FILE\_PATH] = "public\_html/templates/"
- char \* executeScript =0
- struct AmmServer\_Instance \* default\_server =0
- struct AmmServer\_Instance \* admin\_server =0
- struct
- AmmServer\_RequestOverride\_Context GET\_override ={{0}}
- struct AmmServer RH Context stats ={0}
- struct AmmServer RH Context form ={0}
- struct AmmServer RH Context chatbox ={0}
- struct AmmServer\_RH\_Context gps ={0}
- struct AmmServer\_RH\_Context random\_chars ={0}
- struct AmmServer\_RH\_Context executeScriptRC ={0}

- 6.8.1 Macro Definition Documentation
- 6.8.1.1 #define ADMIN\_BINDING\_PORT 8082
- 6.8.1.2 #define DEFAULT\_BINDING\_PORT 8080
- 6.8.1.3 #define ENABLE\_ADMIN\_PAGE 0
- 6.8.1.4 #define ENABLE\_CHAT\_BOX 0
- 6.8.1.5 #define ENABLE\_PASSWORD\_PROTECTION 0
- 6.8.1.6 #define MAX\_BINDING\_PORT 65534
- 6.8.1.7 #define MAX\_SCRIPT\_RESPONSE\_SIZE 40960
- 6.8.2 Function Documentation
- 6.8.2.1 void close\_dynamic\_content ( )

Here is the call graph for this function:

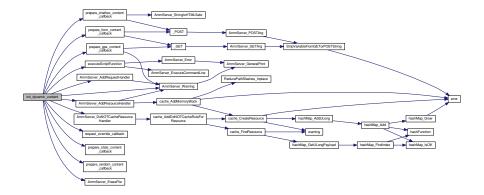


6.8.2.2 void\* executeScriptFunction ( struct AmmServer\_DynamicRequest \* rqst )



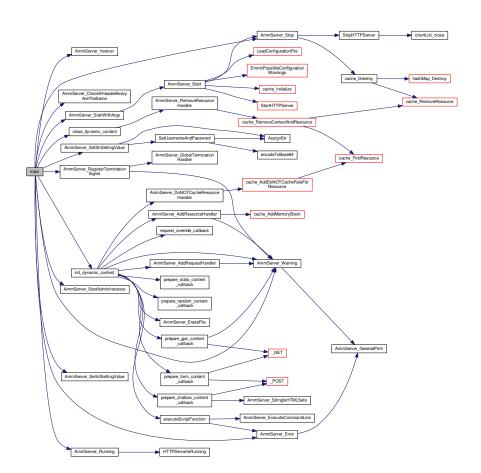
# 6.8.2.3 void init\_dynamic\_content ( )

Here is the call graph for this function:



# 6.8.2.4 int main ( int argc, char \* argv[] )

Dynamic content code ..! END -----



6.8.2.5 void\* prepare\_chatbox\_content\_callback ( struct AmmServer\_DynamicRequest \* rqst )

Here is the call graph for this function:



6.8.2.6 void\* prepare\_form\_content\_callback ( struct AmmServer\_DynamicRequest \* rqst )

Here is the call graph for this function:



6.8.2.7 void\* prepare\_gps\_content\_callback ( struct AmmServer\_DynamicRequest \* rqst )

Here is the call graph for this function:



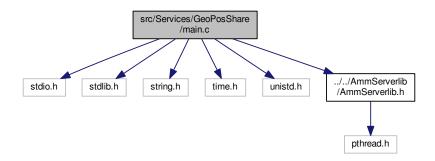
- 6.8.2.8 void\* prepare\_random\_content\_callback ( struct AmmServer\_DynamicRequest \* rqst )
- 6.8.2.9 void\* prepare\_stats\_content\_callback ( struct AmmServer DynamicRequest \* rqst )
- 6.8.2.10 void\* request\_override\_callback ( char \* content )
- 6.8.3 Variable Documentation
- 6.8.3.1 char admin\_root[MAX\_FILE\_PATH] = "admin\_html/"
- 6.8.3.2 struct AmmServer\_Instance\* admin\_server =0
- 6.8.3.3 struct AmmServer\_RH\_Context chatbox ={0}
- 6.8.3.4 struct AmmServer\_Instance\* default\_server =0

Dynamic content code ..! START!

```
6.8.3.5 char* executeScript =0
6.8.3.6 struct AmmServer_RH_Context executeScriptRC ={0}
6.8.3.7 struct AmmServer_RH_Context form ={0}
6.8.3.8 struct AmmServer_RequestOverride_Context GET_override ={{0}}
6.8.3.9 struct AmmServer_RH_Context gps ={0}
6.8.3.10 struct AmmServer_RH_Context random_chars ={0}
6.8.3.11 struct AmmServer_RH_Context stats ={0}
6.8.3.12 char templates_root[MAX_FILE_PATH] ="public_html/templates/"
6.8.3.13 char webserver_root[MAX_FILE_PATH] ="public_html/"
```

# 6.9 src/Services/GeoPosShare/main.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <unistd.h>
#include "../../AmmServerlib/AmmServerlib.h"
Include dependency graph for main.c:
```



# Macros

- #define MAX\_BINDING\_PORT 65534
- #define DEFAULT\_BINDING\_PORT 8081
- #define ADMIN BINDING PORT 8082
- #define ENABLE ADMIN PAGE 0

### **Functions**

- void \* prepare gps content callback (struct AmmServer DynamicRequest \*rqst)
- void \* request\_override\_callback (char \*content)

- void init\_dynamic\_content ()
- void close\_dynamic\_content ()
- int main (int argc, char \*argv[])

### **Variables**

- char admin\_root [MAX\_FILE\_PATH] ="admin\_html/"
- char webserver\_root [MAX\_FILE\_PATH] ="public\_html/"
- char templates root [MAX\_FILE\_PATH] ="public\_html/templates/"
- struct AmmServer\_Instance \* default\_server =0
- struct AmmServer\_RequestOverride\_Context GET\_override ={{0}}
- struct AmmServer\_RH\_Context gps ={0}
- 6.9.1 Macro Definition Documentation
- 6.9.1.1 #define ADMIN\_BINDING\_PORT 8082
- 6.9.1.2 #define DEFAULT\_BINDING\_PORT 8081
- 6.9.1.3 #define ENABLE\_ADMIN\_PAGE 0
- 6.9.1.4 #define MAX\_BINDING\_PORT 65534
- 6.9.2 Function Documentation
- 6.9.2.1 void close\_dynamic\_content ( )

Here is the call graph for this function:



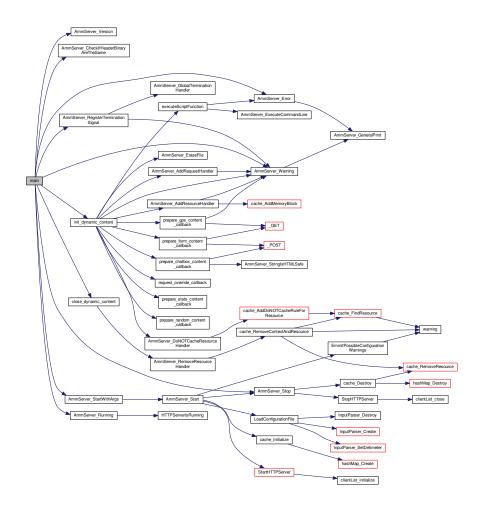
6.9.2.2 void init\_dynamic\_content ( )



6.9.2.3 int main ( int argc, char \* argv[] )

Dynamic content code ..! END -----

Here is the call graph for this function:



6.9.2.4 void\* prepare\_gps\_content\_callback ( struct AmmServer\_DynamicRequest \* rqst )

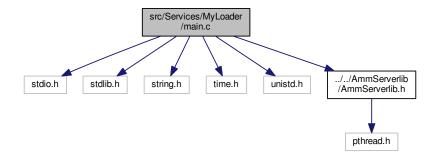


- 6.9.2.5 void\* request\_override\_callback ( char \* content )
- 6.9.3 Variable Documentation
- 6.9.3.1 char admin\_root[MAX\_FILE\_PATH] = "admin\_html/"

- 6.9.3.2 struct AmmServer\_Instance\* default\_server =0
  6.9.3.3 struct AmmServer\_RequestOverride\_Context GET\_override ={{0}}
  6.9.3.4 struct AmmServer\_RH\_Context gps ={0}
  6.9.3.5 char templates\_root[MAX\_FILE\_PATH] ="public\_html/templates/"
- 6.9.3.6 char webserver\_root[MAX\_FILE\_PATH] ="public\_html/"

# 6.10 src/Services/MyLoader/main.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <unistd.h>
#include "../../AmmServerlib/AmmServerlib.h"
Include dependency graph for main.c:
```



#### **Macros**

• #define DEFAULT BINDING PORT 8081

### **Functions**

- void \* prepare\_stats\_content\_callback (struct AmmServer\_DynamicRequest \*rqst)
- void request override callback (void \*request)
- void \* processUploadCallback (struct AmmServer\_DynamicRequest \*rqst)
- void init\_dynamic\_content ()
- void close\_dynamic\_content ()
- int main (int argc, char \*argv[])

## **Variables**

- char webserver root [MAX\_FILE\_PATH] ="src/MyLoader/htmlData/"
- char templates root [MAX\_FILE\_PATH] ="public\_html/templates/"
- struct AmmServer\_Instance \* default\_server =0

- struct AmmServer\_RequestOverride\_Context GET\_override ={{0}}
- struct AmmServer\_RH\_Context uploadProcessor ={0}
- struct AmmServer\_RH\_Context stats ={0}

# 6.10.1 Macro Definition Documentation

6.10.1.1 #define DEFAULT\_BINDING\_PORT 8081

# 6.10.2 Function Documentation

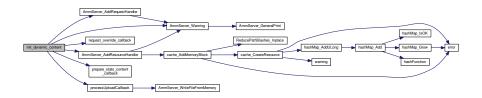
6.10.2.1 void close\_dynamic\_content ( )

Here is the call graph for this function:



## 6.10.2.2 void init\_dynamic\_content()

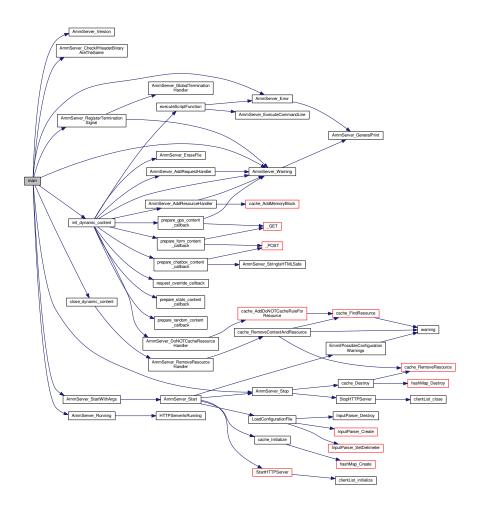
Here is the call graph for this function:



6.10.2.3 int main ( int argc, char \* argv[] )

Dynamic content code ..! END -----

Here is the call graph for this function:



6.10.2.4 void\* prepare\_stats\_content\_callback ( struct AmmServer\_DynamicRequest \* rqst )

6.10.2.5 void\* processUploadCallback ( struct AmmServer\_DynamicRequest \* rqst )



6.10.2.6 void request\_override\_callback ( void \* request )

Here is the call graph for this function:

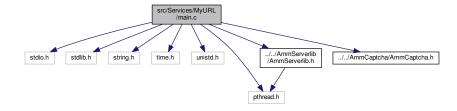


# 6.10.3 Variable Documentation

- 6.10.3.1 struct AmmServer Instance\* default\_server =0
- 6.10.3.2 struct AmmServer\_RequestOverride\_Context GET\_override ={{0}}
- 6.10.3.3 struct AmmServer\_RH\_Context stats ={0}
- 6.10.3.4 char templates\_root[MAX\_FILE\_PATH] = "public\_html/templates/"
- 6.10.3.5 struct AmmServer\_RH\_Context uploadProcessor ={0}
- 6.10.3.6 char webserver\_root[MAX\_FILE\_PATH] ="src/MyLoader/htmlData/"

# 6.11 src/Services/MyURL/main.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <unistd.h>
#include <pthread.h>
#include "../../AmmServerlib/AmmServerlib.h"
#include "../../AmmCaptcha/AmmCaptcha.h"
Include dependency graph for main.c:
```



## **Data Structures**

• struct URLDB

#### **Macros**

- #define ENABLE CAPTCHA SYSTEM 1
- #define USE\_BINARY\_SEARCH 1
- #define MAX BINDING PORT 65534
- #define MAX CAPTCHA JPG SIZE 10 \* 1024
- #define DEFAULT BINDING PORT 8080
- #define DYNAMIC\_PAGES\_MEMORY\_COMMITED 4096
- #define MAX TO SIZE 32
- #define MAX\_LONG\_URL\_SIZE 2048
- #define MAX LINKS 200000
- #define LINK ALLOCATION STEP 5000
- #define REGROUP AFTER X UNSORTED LINKS 1000

#### **Functions**

- int is an unsafe str (char \*input, unsigned int input length)
- int Append2MyURLDBFile (char \*filename, char \*longURL, char \*shortURL)
- unsigned long hashURL (char \*str)
- unsigned int allocateLinksIfNeeded ()
- int isURLDBSorted ()
- int printURLDB ()
- int struct\_cmp\_urldb\_items (const void \*a, const void \*b)
- unsigned int Find longURLSerial (char \*shortURL, int \*found)
- unsigned int Find\_longURL (char \*shortURL, int \*found)
- char \* Get\_longURL (char \*shortURL)
- int ReWriteMyURLDBFile (char \*filename, struct URLDB \*links, unsigned int loaded links)
- int ResortDB (char \*db\_file, struct URLDB \*links, unsigned int loaded\_links)
- unsigned long Add\_MyURL (char \*longURL, char \*shortURL, int saveit)
- int LoadMyURLDBFile (char \*filename)
- void \* serve\_error\_url\_page (struct AmmServer\_DynamicRequest \*rqst)
- void \* serve\_captcha\_page (struct AmmServer\_DynamicRequest \*rqst)
- void \* serve\_create\_url\_page (struct AmmServer\_DynamicRequest \*rqst)
- void \* serve goto url page (struct AmmServer DynamicRequest \*rqst)
- void resolveRequest (void \*request)
- void init dynamic content ()
- · void close dynamic content ()
- int main (int argc, char \*argv[])

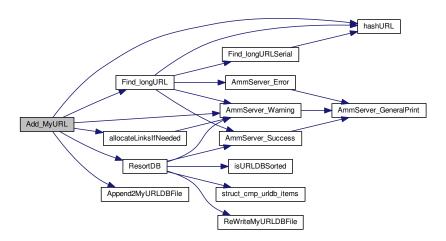
#### **Variables**

- char webserver\_root [MAX\_FILE\_PATH] ="public\_html/"
- char templates\_root [MAX\_FILE\_PATH] = "public\_html/templates/"
- char service\_filename\_noslash [5] ="go"
- char service filename [5] ="/go"
- char service root [128] ="http://myurl.ammar.gr/go"
- char service root withoutfilename [128] = "http://myurl.ammar.gr/"
- char \* default failed = (char\*)"http://myurl.ammar.gr/error.html"
- char db\_file [128] ="myurl.db"
- pthread\_mutex\_t db\_fileLock
- pthread\_mutex\_t db\_addIDLock
- char indexPagePath [128] ="src/Services/MyURL/myurl.html"
- char \* indexPage =0
- unsigned int indexPageLength =0

- struct AmmServer\_Instance \* myurl\_server =0
- struct
   AmmServer\_RequestOverride\_Context requestResolver ={{0}}
- struct AmmServer\_RH\_Context error\_url ={0}
- struct AmmServer\_RH\_Context create\_url ={0}
- struct AmmServer\_RH\_Context goto\_url ={0}
- struct AmmServer\_RH\_Context captcha\_url ={0}
- unsigned int loaded\_links =0
- unsigned int sorted\_links =0
- unsigned int allocated links =0
- struct URLDB \* links =0
- 6.11.1 Macro Definition Documentation
- 6.11.1.1 #define DEFAULT\_BINDING\_PORT 8080
- 6.11.1.2 #define DYNAMIC\_PAGES\_MEMORY\_COMMITED 4096
- 6.11.1.3 #define ENABLE\_CAPTCHA\_SYSTEM 1
- 6.11.1.4 #define LINK\_ALLOCATION\_STEP 5000
- 6.11.1.5 #define MAX\_BINDING\_PORT 65534
- 6.11.1.6 #define MAX\_CAPTCHA\_JPG\_SIZE 10 \* 1024
- 6.11.1.7 #define MAX\_LINKS 200000
- 6.11.1.8 #define MAX\_LONG\_URL\_SIZE 2048
- 6.11.1.9 #define MAX\_TO\_SIZE 32
- 6.11.1.10 #define REGROUP\_AFTER\_X\_UNSORTED\_LINKS 1000
- 6.11.1.11 #define USE\_BINARY\_SEARCH 1
- 6.11.2 Function Documentation

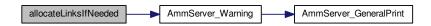
## 6.11.2.1 unsigned long Add\_MyURL ( char \* longURL, char \* shortURL, int saveit )

Here is the call graph for this function:



# 6.11.2.2 unsigned int allocateLinkslfNeeded ( )

Here is the call graph for this function:

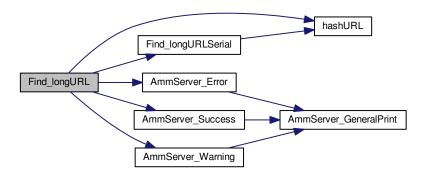


- 6.11.2.3 int Append2MyURLDBFile ( char \* filename, char \* longURL, char \* shortURL )
- 6.11.2.4 void close\_dynamic\_content ( )



6.11.2.5 unsigned int Find\_longURL ( char \* shortURL, int \* found ) [inline]

Here is the call graph for this function:

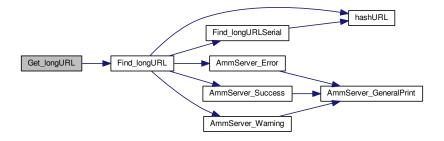


6.11.2.6 unsigned int Find\_longURLSerial ( char \* shortURL, int \* found ) [inline]

Here is the call graph for this function:



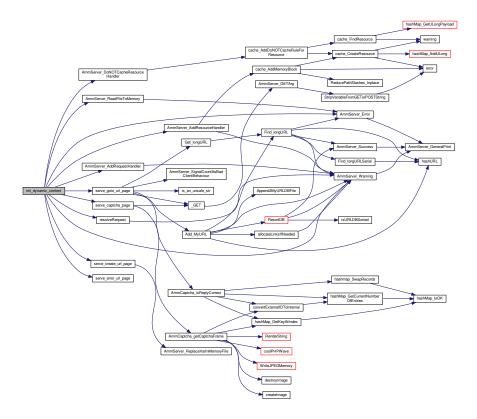
6.11.2.7 char\* Get\_longURL ( char\* shortURL )



6.11.2.8 unsigned long hashURL ( char \* str )

6.11.2.9 void init\_dynamic\_content ( )

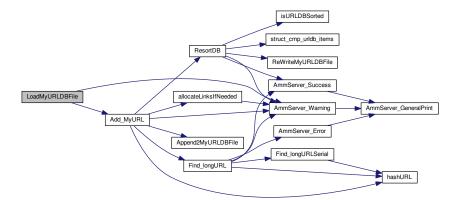
Here is the call graph for this function:



6.11.2.10 int is\_an\_unsafe\_str ( char \* input, unsigned int input\_length )

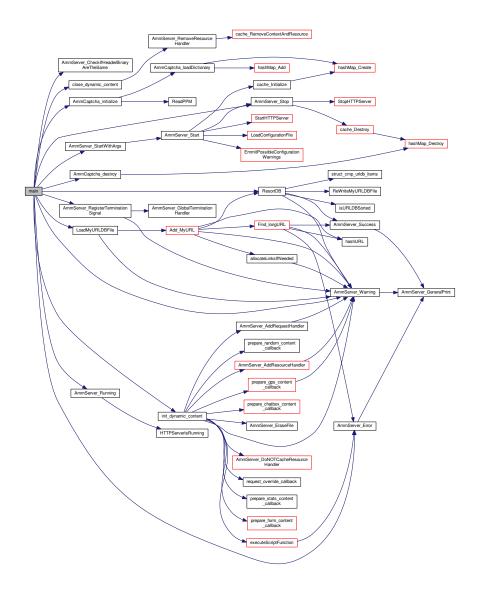
6.11.2.11 int isURLDBSorted ( )

6.11.2.12 int LoadMyURLDBFile ( char \* filename )



# 6.11.2.13 int main ( int argc, char \* argv[] )

Here is the call graph for this function:



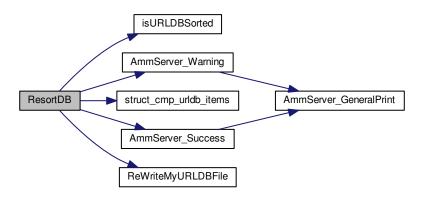
# 6.11.2.14 int printURLDB ( )

# 6.11.2.15 void resolveRequest ( void \* request )



6.11.2.16 int ResortDB ( char \* db\_file, struct URLDB \* links, unsigned int loaded\_links )

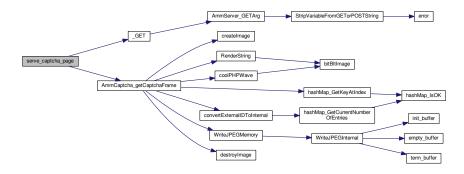
Here is the call graph for this function:



6.11.2.17 int ReWriteMyURLDBFile ( char \* filename, struct URLDB \* links, unsigned int loaded\_links )

6.11.2.18 void\* serve\_captcha\_page ( struct AmmServer\_DynamicRequest \* rqst )

Here is the call graph for this function:

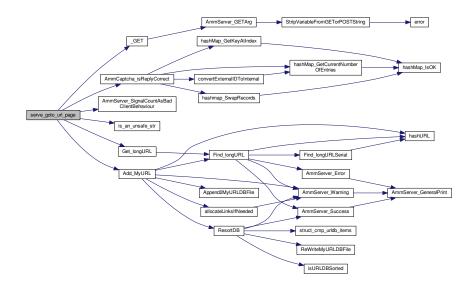


6.11.2.19 void\* serve\_create\_url\_page ( struct AmmServer\_DynamicRequest \* rqst )



```
6.11.2.20 void* serve_error_url_page ( struct AmmServer_DynamicRequest * rqst )
```

6.11.2.21 void\* serve\_goto\_url\_page ( struct AmmServer\_DynamicRequest \* rqst )

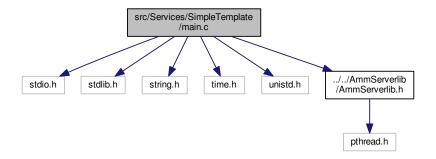


- 6.11.2.22 int struct\_cmp\_urldb\_items ( const void \* a, const void \* b )
- 6.11.3 Variable Documentation
- 6.11.3.1 unsigned int allocated\_links =0
- 6.11.3.2 struct AmmServer\_RH\_Context captcha\_url ={0}
- 6.11.3.3 struct AmmServer\_RH\_Context create\_url ={0}
- 6.11.3.4 pthread\_mutex\_t db\_addIDLock
- 6.11.3.5 char db\_file[128] ="myurl.db"
- 6.11.3.6 pthread\_mutex\_t db\_fileLock
- 6.11.3.7 char\* default\_failed = (char\*)"http://myurl.ammar.gr/error.html"
- 6.11.3.8 struct AmmServer\_RH\_Context error\_url ={0}
- 6.11.3.9 struct AmmServer\_RH\_Context goto\_url ={0}
- 6.11.3.10 char\* indexPage =0
- 6.11.3.11 unsigned int indexPageLength =0
- 6.11.3.12 char indexPagePath[128] ="src/Services/MyURL/myurl.html"

```
6.11.3.13 struct URLDB* links =0
6.11.3.14 unsigned int loaded_links =0
6.11.3.15 struct AmmServer_Instance* myurl_server =0
6.11.3.16 struct AmmServer_RequestOverride_Context requestResolver ={{0}}
6.11.3.17 char service_filename[5] = "/go"
6.11.3.18 char service_filename_noslash[5] = "go"
6.11.3.19 char service_root[128] = "http://myurl.ammar.gr/go"
6.11.3.20 char service_root_withoutfilename[128] = "http://myurl.ammar.gr/"
6.11.3.21 unsigned int sorted_links =0
6.11.3.22 char templates_root[MAX_FILE_PATH] = "public_html/templates/"
6.11.3.23 char webserver_root[MAX_FILE_PATH] = "public_html/"
```

# 6.12 src/Services/SimpleTemplate/main.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <unistd.h>
#include "../../AmmServerlib/AmmServerlib.h"
Include dependency graph for main.c:
```



### **Macros**

• #define DEFAULT BINDING PORT 8080

# **Functions**

void \* prepare\_stats\_content\_callback (struct AmmServer\_DynamicRequest \*rqst)

- void \* prepare\_random\_content\_callback (struct AmmServer\_DynamicRequest \*rqst)
- void request\_override\_callback (void \*request)
- void init\_dynamic\_content ()
- · void close\_dynamic\_content ()
- int main (int argc, char \*argv[])

#### **Variables**

- char webserver\_root [MAX\_FILE\_PATH] ="public\_html/"
- char templates\_root [MAX\_FILE\_PATH] ="public\_html/templates/"
- struct AmmServer\_Instance \* default\_server =0
- struct

AmmServer\_RequestOverride\_Context GET\_override ={{0}}

- struct AmmServer\_RH\_Context random\_chars ={0}
- struct AmmServer\_RH\_Context stats ={0}

#### 6.12.1 Macro Definition Documentation

6.12.1.1 #define DEFAULT\_BINDING\_PORT 8080

### 6.12.2 Function Documentation

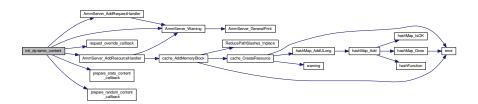
6.12.2.1 void close\_dynamic\_content()

Here is the call graph for this function:



#### 6.12.2.2 void init\_dynamic\_content ( )

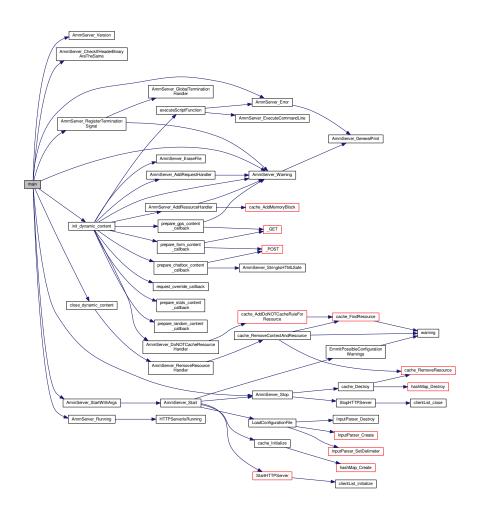
Here is the call graph for this function:



6.12.2.3 int main (int argc, char \* argv[])

Dynamic content code ..! END -----

Here is the call graph for this function:



- 6.12.2.4 void\* prepare\_random\_content\_callback ( struct AmmServer\_DynamicRequest \* rqst )
- 6.12.2.5 void\* prepare\_stats\_content\_callback ( struct AmmServer\_DynamicRequest \* rqst )
- 6.12.2.6 void request\_override\_callback ( void \* request )
- 6.12.3 Variable Documentation
- 6.12.3.1 struct AmmServer\_Instance\* default\_server =0

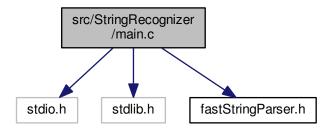
Dynamic content code ..! START!

- 6.12.3.2 struct AmmServer\_RequestOverride\_Context GET\_override ={{0}}
- 6.12.3.3 struct AmmServer\_RH\_Context random\_chars ={0}
- 6.12.3.4 struct AmmServer\_RH\_Context stats ={0}
- 6.12.3.5 char templates\_root[MAX\_FILE\_PATH] = "public\_html/templates/"

6.12.3.6 char webserver\_root[MAX\_FILE\_PATH] = "public\_html/"

# 6.13 src/StringRecognizer/main.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include "fastStringParser.h"
Include dependency graph for main.c:
```



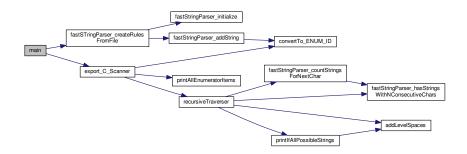
### **Functions**

• int main (int argc, char \*argv[])

## 6.13.1 Function Documentation

6.13.1.1 int main ( int argc, char \* argv[] )

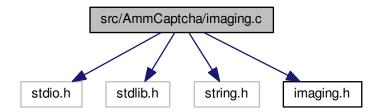
Here is the call graph for this function:



# 6.14 src/AmmCaptcha/imaging.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "imaging.h"
```

Include dependency graph for imaging.c:



#### **Macros**

- #define READ\_CREATES\_A\_NEW\_PIXEL\_BUFFER 1
- #define PPMREADBUFLEN 256
- #define DISPLAY DEBUG INFO 0

#### **Functions**

- struct Image \* createImage (unsigned int width, unsigned int height, unsigned int depth)
- struct Image \* copyImage (struct Image \*source)
- int destroyImage (struct Image \*source)
- int bitBltImage (struct Image \*target, unsigned int targetX, unsigned int targetY, struct Image \*source, unsigned int sourceX, unsigned int sourceY, unsigned int width, unsigned int height)
- int bitBltImageRotated (struct Image \*target, unsigned int targetCenterX, unsigned int targetCenterY, float rotation, struct Image \*source, unsigned int sourceX, unsigned int sourceY, unsigned int width, unsigned int height)
- int ReadPPM (struct Image \*pic, char \*filename, char read\_only\_header)
- int WritePPM (struct Image \*pic, char \*filename)

#### 6.14.1 Macro Definition Documentation

- 6.14.1.1 #define DISPLAY\_DEBUG\_INFO 0
- 6.14.1.2 #define PPMREADBUFLEN 256
- 6.14.1.3 #define READ\_CREATES\_A\_NEW\_PIXEL\_BUFFER 1

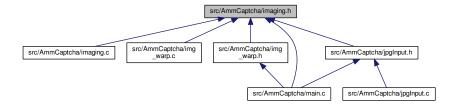
#### 6.14.2 Function Documentation

- 6.14.2.1 int bitBltImage ( struct Image \* target, unsigned int targetX, unsigned int targetY, struct Image \* source, unsigned int sourceX, unsigned int sourceY, unsigned int width, unsigned int height )
- 6.14.2.2 int bitBltImageRotated ( struct Image \* target, unsigned int targetCenterX, unsigned int targetCenterY, float rotation, struct Image \* source, unsigned int sourceX, unsigned int sourceY, unsigned int width, unsigned int height )
- 6.14.2.3 struct Image \* copylmage ( struct Image \* source )

- 6.14.2.4 struct Image\* createImage ( unsigned int width, unsigned int height, unsigned int depth )
  6.14.2.5 int destroyImage ( struct Image \* source )
  6.14.2.6 int ReadPPM ( struct Image \* pic, char \* filename, char read\_only\_header )
- 6.15 src/AmmCaptcha/imaging.h File Reference

6.14.2.7 int WritePPM ( struct Image \* pic, char \* filename )

This graph shows which files directly or indirectly include this file:



#### **Data Structures**

struct Image

### **Functions**

- struct Image \* createImage (unsigned int width, unsigned int height, unsigned int depth)
- struct Image \* copyImage (struct Image \*source)
- int destroyImage (struct Image \*source)
- int bitBltImage (struct Image \*target, unsigned int targetX, unsigned int targetY, struct Image \*source, unsigned int sourceX, unsigned int sourceY, unsigned int width, unsigned int height)
- int ReadPPM (struct Image \*pic, char \*filename, char read only header)
- int WritePPM (struct Image \*pic, char \*filename)

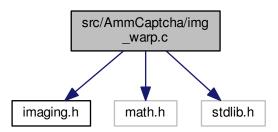
# 6.15.1 Function Documentation

- 6.15.1.1 int bitBltImage ( struct Image \* target, unsigned int targetX, unsigned int targetY, struct Image \* source, unsigned int sourceX, unsigned int sourceY, unsigned int width, unsigned int height )
- 6.15.1.2 struct Image \* copylmage ( struct Image \* source )
- 6.15.1.3 struct Image\* createImage ( unsigned int width, unsigned int height, unsigned int depth )
- 6.15.1.4 int destroylmage ( struct Image \* source )
- 6.15.1.5 int ReadPPM ( struct Image \* pic, char \* filename, char read\_only\_header )
- 6.15.1.6 int WritePPM ( struct Image \* pic, char \* filename )

# 6.16 src/AmmCaptcha/img\_warp.c File Reference

```
#include "imaging.h"
#include <math.h>
#include <stdlib.h>
```

Include dependency graph for img\_warp.c:



#### **Macros**

- #define ABS(num1) ( (num1) >=0 ? (num1) : (-1\*num1) )
- #define ABSDIFF(num1, num2) ( (num1-num2) >=0 ? (num1-num2) : (num2 num1) )

#### **Functions**

- int warpImage (struct Image \*target, unsigned int posX, unsigned int posY, signed int warpDeltaX, signed int warpDeltaY)
- int coolPHPWave (struct Image \*target, unsigned int periodX, unsigned int periodY, signed int amplitudeX, signed int amplitudeY)

## 6.16.1 Macro Definition Documentation

```
6.16.1.1 #define ABS( num1 ) ( (num1) >=0 ? (num1) : (-1*num1) )
```

6.16.1.2 #define ABSDIFF( num1, num2) ((num1-num2) >=0 ? (num1-num2) : (num2 - num1))

# 6.16.2 Function Documentation

6.16.2.1 int coolPHPWave ( struct Image \* target, unsigned int periodX, unsigned int periodY, signed int amplitudeX, signed int amplitudeY)

This is a C version of the PHP script from Jose Rodriguez , currently used as a swirling mechanism , it is GPLv3 as this project :)

Author

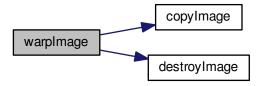
Jose Rodriguez jose.rodriguez@exec.cl GPLv3 captcha 0.3

Here is the call graph for this function:



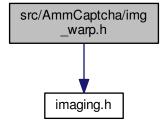
6.16.2.2 int warpImage ( struct Image \* target, unsigned int posX, unsigned int posY, signed int warpDeltaX, signed int warpDeltaX)

Here is the call graph for this function:

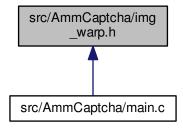


# 6.17 src/AmmCaptcha/img\_warp.h File Reference

#include "imaging.h"
Include dependency graph for img\_warp.h:



This graph shows which files directly or indirectly include this file:



### **Functions**

- int warpImage (struct Image \*target, unsigned int posX, unsigned int posY, signed int warpDeltaX, signed int warpDeltaY)
- int coolPHPWave (struct Image \*target, unsigned int periodX, unsigned int periodY, signed int amplitudeX, signed int amplitudeY)

#### 6.17.1 Function Documentation

6.17.1.1 int coolPHPWave ( struct Image \* target, unsigned int periodX, unsigned int periodY, signed int amplitudeX, signed int amplitudeY)

This is a C version of the PHP script from Jose Rodriguez , currently used as a swirling mechanism , it is GPLv3 as this project :)

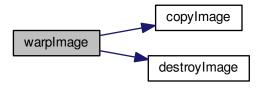
#### Author

Jose Rodriguez jose.rodriguez@exec.cl GPLv3 captcha 0.3



6.17.1.2 int warpImage ( struct Image \* target, unsigned int posX, unsigned int posY, signed int warpDeltaX, signe

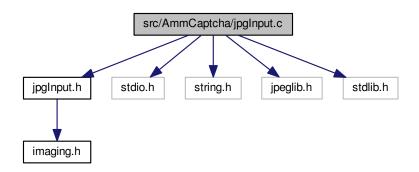
Here is the call graph for this function:



# 6.18 src/AmmCaptcha/jpgInput.c File Reference

```
#include "jpgInput.h"
#include <stdio.h>
#include <string.h>
#include <jpeglib.h>
#include <stdlib.h>
```

Include dependency graph for jpgInput.c:



#### **Functions**

- void init\_buffer (struct jpeg\_compress\_struct \*cinfo)
- int empty\_buffer (struct jpeg\_compress\_struct \*cinfo)
- void term\_buffer (struct jpeg\_compress\_struct \*cinfo)
- int fastJPGHeaderCheck (FILE \*file)
- int ReadJPEG (char \*filename, struct Image \*pic, char read\_only\_header)
- int WriteJPEGInternal (char \*filename, struct Image \*pic, char \*mem, unsigned long \*mem\_size)
- int WriteJPEGFile (struct Image \*pic, char \*filename)
- int WriteJPEGMemory (struct Image \*pic, char \*mem, unsigned long \*mem\_size)
- int jpegtest ()

### 6.18.1 Function Documentation

- 6.18.1.1 int empty\_buffer ( struct jpeg\_compress\_struct \* cinfo )
- 6.18.1.2 int fastJPGHeaderCheck (FILE \* file )
- 6.18.1.3 void init\_buffer ( struct jpeg\_compress\_struct \* cinfo )

read\_jpeg\_file Reads from a jpeg file on disk specified by filename and saves into the raw\_image buffer in an uncompressed format.

#### Returns

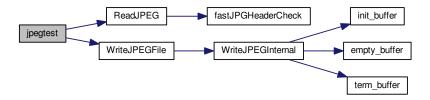
positive integer if successful, -1 otherwise

#### **Parameters**

*filename	char string specifying the file name to read from

# 6.18.1.4 int jpegtest ( )

Here is the call graph for this function:



6.18.1.5 int ReadJPEG ( char \* filename, struct Image \* pic, char read\_only\_header )

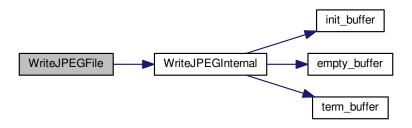
Here is the call graph for this function:



6.18.1.6 void term\_buffer ( struct jpeg\_compress\_struct \* cinfo )

### 6.18.1.7 int WriteJPEGFile ( struct Image \* pic, char \* filename )

Here is the call graph for this function:



6.18.1.8 int WriteJPEGInternal ( char \* filename, struct Image \* pic, char \* mem, unsigned long \* mem\_size )

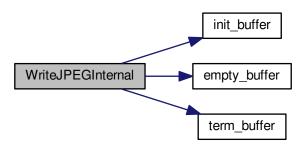
write\_jpeg\_file Writes the raw image data stored in the raw\_image buffer to a jpeg image with default compression and smoothing options in the file specified by \*filename.

### Returns

positive integer if successful, -1 otherwise

#### **Parameters**

*filename	char string specifying the file name to save to
-----------	---



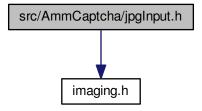
6.18.1.9 int WriteJPEGMemory ( struct Image \* pic, char \* mem, unsigned long \* mem\_size )

Here is the call graph for this function:

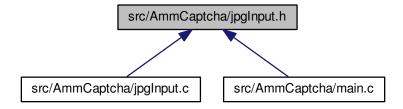


# 6.19 src/AmmCaptcha/jpgInput.h File Reference

#include "imaging.h"
Include dependency graph for jpgInput.h:



This graph shows which files directly or indirectly include this file:



### **Macros**

• #define USE\_JPG\_FILES 1

### **Functions**

- int ReadJPEG (char \*filename, struct Image \*pic, char read\_only\_header)
- int WriteJPEGFile (struct Image \*pic, char \*filename)
- int WriteJPEGMemory (struct Image \*pic, char \*mem, unsigned long \*mem\_size)

### 6.19.1 Macro Definition Documentation

6.19.1.1 #define USE\_JPG\_FILES 1

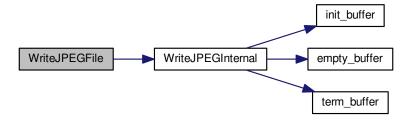
### 6.19.2 Function Documentation

6.19.2.1 int ReadJPEG ( char \* filename, struct Image \* pic, char read\_only\_header )

Here is the call graph for this function:

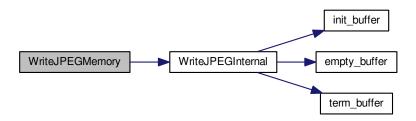


6.19.2.2 int WriteJPEGFile ( struct Image \* pic, char \* filename )



6.19.2.3 int WriteJPEGMemory ( struct Image \* pic, char \* mem, unsigned long \* mem\_size )

Here is the call graph for this function:

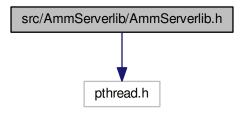


### 6.20 src/AmmServerlib/AmmServerlib.h File Reference

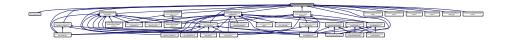
The Main Header for AmmarServer.

#include <pthread.h>

Include dependency graph for AmmServerlib.h:



This graph shows which files directly or indirectly include this file:



### **Data Structures**

• struct HTTPHeader

Each HTTP Request has a header, this is the internal structure that carries the information about the header of an HTTP request parsed and ready for easy for consumption by the various consumers of HTTP requests.

• struct AmmServer\_RequestOverride\_Context

We can override/intercept connections before the very fundamental HTTP stage using a request override context and AmmServer\_AddRequestHandler This is the structure that holds the information and what to be called back to populate the response.

• struct AmmServer\_DynamicRequest

When a call to a function that is a dynamic request is done this is the structure that holds the information.

struct AmmServer RH Context

We can override resources to respond with our own C function code, to do so a AmmServer\_DynamicRequest must be populated using a AmmServer\_AddResourceHandler.

· struct AmmServer Instance Settings

Each Instance of AmmarServer has some basic settings, which are stored in AmmServer\_Instance\_Settings.

struct AmmServer Instance

This holds all the information about an Ammar Server Instance, sockets, thread pools, cache, memory, settings etc, this is the central structure for holding context.

struct HTTPTransaction

Structure to keep data for an HTTP Transaction.

#### **Macros**

#define AMMAR SERVER HTTP HEADER SPEC 125

An enumerator that lists the types of requests , per HTTP spec , see http://www.w3.org/-Protocols/rfc2616/rfc2616-sec9.html Of course not all of them are supported/used internally but they are listed in the same order to maintain spec compatibility.

- #define MAX\_IP\_STRING\_SIZE 32
- #define MAX\_QUERY 2048
- #define MAX\_RESOURCE 2048
- #define MAX FILE PATH 1024
- #define POPEN BUFFER SIZE 256

Size for popen replies.

• #define MAX\_INSTANCE\_NAME\_STRING 128

#### **Enumerations**

enum TypesOfRequests {
 NONE =0, HEAD, GET, POST,
 PUT, DELETE, TRACE, OPTIONS,
 CONNECT, PATCH, BAD }

enum RHScenarios { SAME PAGE FOR ALL CLIENTS = 0, DIFFERENT PAGE FOR EACH CLIENT }

Each Dynamic Resource Handler can have multiple profiles for optimizing performance/memory usage etc. For now there are 2 profiles/scenarios. The first one is where there is a global state that all clients should share The second one is where there is a different page for each client, which is more memory intensive since there are separate buffers etc for each request.

• enum AmmServInfos { AMMINF\_ACTIVE\_CLIENTS =0, AMMINF\_ACTIVE\_THREADS }

Enumerator for calls AmmServer\_GetInfo.

• enum AmmServSettings { AMMSET\_PASSWORD\_PROTECTION =0, AMMSET\_TEST }

Enumerator for calls AmmServer\_GetIntSettingValue and AmmServer\_SetIntSettingValue.

 enum AmmServStrSettings { AMMSET\_USERNAME\_STR =0, AMMSET\_PASSWORD\_STR, AMMSET\_T-ESTSTR }

Enumerator for calls AmmServer\_GetStrSettingValue and AmmServer\_SetStrSettingValue.

#### **Functions**

char \* AmmServer Version ()

Returns a string with the version of AmmarServer, in case it returns NULL it means that we are linked to AmmarServerNULL which means a fake binary.

int AmmServer\_CheckIfHeaderBinaryAreTheSame (int headerSpec)

Internal Check to compare against changes of the header files.

void AmmServer Warning (const char \*format,...)

Writes the C string pointed by format to stderr, as a warning (Yellow) and logs it to the appropriate log If format includes format specifiers (subsequences beginning with %), the additional arguments following format are formatted and inserted in the resulting string replacing their respective specifiers.

void AmmServer Error (const char \*format,...)

Writes the C string pointed by format to stderr, as an error (Red) and logs it to the appropriate log If format includes format specifiers (subsequences beginning with %), the additional arguments following format are formatted and inserted in the resulting string replacing their respective specifiers.

void AmmServer Success (const char \*format,...)

Writes the C string pointed by format to stderr, as a success ( Green ) and logs it to the appropriate log If format includes format specifiers (subsequences beginning with %), the additional arguments following format are formatted and inserted in the resulting string replacing their respective specifiers.

• struct AmmServer\_Instance \* AmmServer\_Start (char \*name, char \*ip, unsigned int port, char \*conf\_file, char \*web\_root\_path, char \*templates\_root\_path)

Start a Web Server, allocate memory, bind ports and return its instance..

• struct AmmServer\_Instance \* AmmServer\_StartWithArgs (char \*name, int argc, char \*\*argv, char \*ip, unsigned int port, char \*conf file, char \*web root path, char \*templates root path)

Start a Web Server, allocate memory, bind ports and return its instance, also process arguments (argc and argv from int main(int argc, char \*argv[]))..

• int AmmServer\_Stop (struct AmmServer\_Instance \*instance)

Stop a Web Server, deallocate memory, free ports and free the server instance..

• int AmmServer Running (struct AmmServer Instance \*instance)

Query if an instance of AmmarServer is initialized and running.

 int AmmServer\_AddRequestHandler (struct AmmServer\_Instance \*instance, struct AmmServer\_Request-Override\_Context \*RequestOverrideContext, char \*request\_type, void \*callback)

Add a request handler to handle requests, before they get processed internally Calling this will bind a C function that will be called and produce output when someone asks for any resource using the specified method TODO: Improve this documenatation.

• int AmmServer\_AddResourceHandler (struct AmmServer\_Instance \*instance, struct AmmServer\_RH\_-Context \*context, char \*resource\_name, char \*web\_root, unsigned int allocate\_mem\_bytes, unsigned int callback\_every\_x\_msec, void \*callback, unsigned int scenario)

Add a request handler to handle dynamic requests , the core mechanic of AmmarServer Calling this will bind a C function that will be called and produce output when someone asks for a resource TODO: Improve this documenatation.

 int AmmServer\_RemoveResourceHandler (struct AmmServer\_Instance \*instance, struct AmmServer\_RH\_-Context \*context, unsigned char free mem)

Remove a request handler that hanles dynamic requests.

• int AmmServer\_GetInfo (struct AmmServer\_Instance \*instance, unsigned int info\_type)

Get an Integer out of the state of an instance , of course one can dive into the instance structure but this is a much more clean way to do this.

int AmmServer GetIntSettingValue (struct AmmServer Instance \*instance, unsigned int set type)

Get an Integer out of the state of an instance, of course one can dive into the instance structure but this is a much more clean way to do this.

int AmmServer\_SetIntSettingValue (struct AmmServer\_Instance \*instance, unsigned int set\_type, int set\_value)

Set an Integer inside the state of an instance , of course one can dive into the instance structure but this is a much more clean way to do this.

• char \* AmmServer GetStrSettingValue (struct AmmServer Instance \*instance, unsigned int set type)

Get a String out of the state of an instance, of course one can dive into the instance structure but this is a much more clean way to do this.

int AmmServer\_SetStrSettingValue (struct AmmServer\_Instance \*instance, unsigned int set\_type, char \*set\_value)

Set an string inside the state of an instance , of course one can dive into the instance structure but this is a much more clean way to do this.

• int AmmServer\_POSTArg (struct AmmServer\_Instance \*instance, struct AmmServer\_DynamicRequest \*rqst, char \*var id IN, char \*var value OUT, unsigned int max var value OUT)

Get a POST argument.

• int AmmServer\_GETArg (struct AmmServer\_Instance \*instance, struct AmmServer\_DynamicRequest \*rqst, char \*var\_id\_IN, char \*var\_value\_OUT, unsigned int max\_var\_value\_OUT)

Get a GET argument.

• int AmmServer\_FILES (struct AmmServer\_Instance \*instance, struct AmmServer\_DynamicRequest \*rqst, char \*var\_id\_IN, char \*var\_value\_OUT, unsigned int max\_var\_value\_OUT)

Access a FILE submitted by a dynamic requested.

• int \_POST (struct AmmServer\_Instance \*instance, struct AmmServer\_DynamicRequest \*rqst, char \*var\_id\_IN, char \*var\_value\_OUT, unsigned int max\_var\_value\_OUT)

Shorthand/Shortcut for AmmServer POSTArg()

int \_GET (struct AmmServer\_Instance \*instance, struct AmmServer\_DynamicRequest \*rqst, char \*var\_id\_I-N, char \*var\_value\_OUT, unsigned int max\_var\_value\_OUT)

Shorthand/Shortcut for AmmServer\_GETArg()

int \_FILES (struct AmmServer\_Instance \*instance, struct AmmServer\_DynamicRequest \*rqst, char \*var\_id-IN, char \*var value OUT, unsigned int max var value OUT)

Shorthand/Shortcut for AmmServer\_FILES()

int AmmServer\_SignalCountAsBadClientBehaviour (struct AmmServer\_Instance \*instance, struct AmmServer\_DynamicRequest \*rqst)

Staged way to easily handle bad clients etc from the clients, currently a stub..!

int AmmServer\_SaveDynamicRequest (char \*filename, struct AmmServer\_Instance \*instance, struct AmmServer\_DynamicRequest \*rqst)

Save Dynamic Request to file.

int AmmServer\_DoNOTCacheResourceHandler (struct AmmServer\_Instance \*instance, struct AmmServer\_RH\_Context \*context)

Set resource handler to no-cache mode, this means whoever asks for it will never get a cached response.

• int AmmServer\_DoNOTCacheResource (struct AmmServer\_Instance \*instance, char \*resource\_name)

Set resource to no-cache mode , this means whoever asks for it will never get a cached response.

struct AmmServer Instance \* AmmServer StartAdminInstance (char \*ip, unsigned int port)

Planned functionality for a default http administrator panel per server per instance, currently not implemented correctly.

• int AmmServer SelfCheck (struct AmmServer Instance \*instance)

Perform a sanity check on the instance of AmmarServer, this is mostly a dev debug tool and an entry point for code inside AmmServerlib.

 int AmmServer\_ExecuteCommandLine (char \*command, char \*what2GetBack, unsigned int what2GetBack-MaxSize)

Execute a command and copy its output to the provided buffer.

int AmmServer\_ReplaceVarInMemoryFile (char \*page, unsigned int pageLength, char \*var, char \*value)

Hot-Replace a variable inside a memory block, typically used to replace placeholders inside text files, like \$\$\$\$\$\$\$NAME\$\$\$\$\$\$\$, the value should be smaller or equal to the var beeing replaced.

char \* AmmServer\_ReadFileToMemory (char \*filename, unsigned int \*length)

Read a file and store it to a freshly allocated memory block.

int AmmServer\_WriteFileFromMemory (char \*filename, char \*memory, unsigned int memoryLength)

Dump a memory block to a file.

int AmmServer\_RegisterTerminationSignal (void \*callback)

Register a function to call a function that gracefully terminates a client when a SIGKILL or the time to stop the server comes.

int AmmServer\_DirectoryExists (char \*filename)

Check if directory Exists.

int AmmServer FileExists (char \*filename)

Check if file Exists.

• int AmmServer\_EraseFile (char \*filename)

Erase a File.

unsigned int AmmServer StringIsHTMLSafe (char \*str)

Check if a string has html elements inside it, so if we append it to a web site we won't have html injected.

### 6.20.1 Detailed Description

The Main Header for AmmarServer. Any application that may want to interface with AmmarServer will probably want to link to libAmmarServer.a and include this header. It provides the entry point for setting up a web share and access to sub-modules on runtime.

**Author** 

Ammar Qammaz (AmmarkoV)

Bug AmmarServer is not properly pentested yet

#### 6.20.2 Macro Definition Documentation

#### 6.20.2.1 #define AMMAR\_SERVER\_HTTP\_HEADER\_SPEC 125

An enumerator that lists the types of requests , per HTTP spec , see http://www.w3.org/-Protocols/rfc2616/rfc2616-sec9.html Of course not all of them are supported/used internally but they are listed in the same order to maintain spec compatibility.

Bug A potential bug might arise if the specs of the header file are changed and someone is linking with an older version libAmmServer.a thats why this value exists

```
6.20.2.2 #define MAX_FILE_PATH 1024
```

6.20.2.3 #define MAX\_INSTANCE\_NAME\_STRING 128

6.20.2.4 #define MAX\_IP\_STRING\_SIZE 32

6.20.2.5 #define MAX\_QUERY 2048

6.20.2.6 #define MAX\_RESOURCE 2048

6.20.2.7 #define POPEN\_BUFFER\_SIZE 256

Size for popen replies.

#### 6.20.3 Enumeration Type Documentation

### 6.20.3.1 enum AmmServInfos

Enumerator for calls AmmServer\_GetInfo.

#### Enumerator

```
AMMINF_ACTIVE_CLIENTS
AMMINF_ACTIVE_THREADS
```

#### 6.20.3.2 enum AmmServSettings

Enumerator for calls AmmServer\_GetIntSettingValue and AmmServer\_SetIntSettingValue.

#### **Enumerator**

```
AMMSET_PASSWORD_PROTECTION
AMMSET_TEST
```

#### 6.20.3.3 enum AmmServStrSettings

Enumerator for calls AmmServer\_GetStrSettingValue and AmmServer\_SetStrSettingValue.

#### **Enumerator**

```
AMMSET_USERNAME_STR
AMMSET_PASSWORD_STR
AMMSET_TESTSTR
```

#### 6.20.3.4 enum RHScenarios

Each Dynamic Resource Handler can have multiple profiles for optimizing performance/memory usage etc. For now there are 2 profiles/scenarios. The first one is where there is a global state that all clients should share The second one is where there is a different page for each client , which is more memory intensive since there are separate buffers etc for each request.

#### Enumerator

```
SAME_PAGE_FOR_ALL_CLIENTS
DIFFERENT_PAGE_FOR_EACH_CLIENT
```

### 6.20.3.5 enum TypesOfRequests

An enumerator that lists the types of requests , per HTTP spec , see http://www.w3.org/-Protocols/rfc2616/rfc2616-sec9.html Of course not all of them are supported/used internally but they are listed in the same order to maintain spec compatibility.

### Enumerator

NONE
HEAD
GET
POST
PUT
DELETE
TRACE
OPTIONS
CONNECT
PATCH

BAD

### 6.20.4 Function Documentation

6.20.4.1 int \_FILES ( struct AmmServer\_Instance \* instance, struct AmmServer\_DynamicRequest \* rqst, char \* var\_id\_IN, char \* var\_value\_OUT, unsigned int max\_var\_value\_OUT )

Shorthand/Shortcut for AmmServer FILES()

Here is the call graph for this function:



6.20.4.2 int \_GET ( struct AmmServer\_Instance \* instance, struct AmmServer\_DynamicRequest \* rqst, char \* var\_id\_IN, char \* var\_value\_OUT, unsigned int max\_var\_value\_OUT )

Shorthand/Shortcut for AmmServer\_GETArg()

Here is the call graph for this function:



6.20.4.3 int \_POST ( struct AmmServer\_Instance \* instance, struct AmmServer\_DynamicRequest \* rqst, char \* var\_id\_IN, char \* var\_value\_OUT, unsigned int max\_var\_value\_OUT)

Shorthand/Shortcut for AmmServer POSTArg()

Here is the call graph for this function:



6.20.4.4 int AmmServer\_AddRequestHandler ( struct AmmServer\_Instance \* instance, struct AmmServer\_RequestOverride\_Context \* RequestOverrideContext, char \* request\_type, void \* callback )

Add a request handler to handle requests , before they get processed internally Calling this will bind a C function that will be called and produce output when someone asks for any resource using the specified method TODO : Improve this documenatation.

#### **Parameters**

An	AmmarServer Instance
Α	AmmServer_RequestOverride_Context to be populated
Request	Туре
Pointer	to function callback

#### Return values

1=Success,0:	Fail

Here is the call graph for this function:



6.20.4.5 int AmmServer\_AddResourceHandler ( struct AmmServer\_Instance \* instance, struct AmmServer\_RH\_Context \* context, char \* resource\_name, char \* web\_root, unsigned int allocate\_mem\_bytes, unsigned int callback\_every\_x\_msec, void \* callback, unsigned int scenario )

Add a request handler to handle dynamic requests , the core mechanic of AmmarServer Calling this will bind a C function that will be called and produce output when someone asks for a resource TODO : Improve this documenatation.

#### **Parameters**

An	AmmarServer Instance
An	AmmServer_RH_Context to be populated
Name	of resource that should get dynamic responses ( i.e. "index.html" )
Root	Path for the specific resource
Memory	chunk to allocate for responses , ( this is the max response size )
Minimum	time between two calls of the function (0 = no minimum time)
Function	to be called and provides output when someone asks for resource
Scenario/Profile	of this resource ( see RHScenarios )

### Return values

1=Success,0=Fail	

Here is the call graph for this function:



6.20.4.6 int AmmServer\_CheckIfHeaderBinaryAreTheSame ( int headerSpec )

Internal Check to compare against changes of the header files.

#### **Parameters**

Header	(should be AMMAR SERVER HTTP HEADER SPEC)	
i icauci	( SHOULD BE AIVINIALL BELLVELL THAT TIEADELL BI EO )	

#### Return values

```
1=Success,0=Failure
```

6.20.4.7 int AmmServer\_DirectoryExists ( char \* filename )

Check if directory Exists.

#### **Parameters**

Path	to directory
------	--------------

#### Return values

```
1=Exists,0=Does not Exist
```

Here is the call graph for this function:



6.20.4.8 int AmmServer\_DoNOTCacheResource ( struct AmmServer\_Instance \* instance, char \* resource\_name )

Set resource to no-cache mode, this means whoever asks for it will never get a cached response.

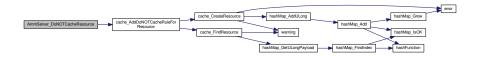
### **Parameters**

Instance	of an AmmarServer
Resource	name that we want to always serve fresh

#### Return values

```
1=Success,0=Failure
```

Here is the call graph for this function:



6.20.4.9 int AmmServer\_DoNOTCacheResourceHandler ( struct AmmServer\_Instance \* instance, struct AmmServer\_RH\_Context \* context )

Set resource handler to no-cache mode, this means whoever asks for it will never get a cached response.

#### **Parameters**

Instance	of an AmmarServer
Resource	context that should always be served fresh ( AmmServer_RH_Context )

#### Return values

1=Success,0=Failure	

Here is the call graph for this function:



6.20.4.10 int AmmServer\_EraseFile ( char \* filename )

Erase a File.

#### **Parameters**

Path	to file

#### Return values

```
1=Success,0=Failure
```

6.20.4.11 void AmmServer\_Error ( const char \* format, ... )

Writes the C string pointed by format to stderr , as an error ( Red ) and logs it to the appropriate log If format includes format specifiers (subsequences beginning with %), the additional arguments following format are formatted and inserted in the resulting string replacing their respective specifiers.

### **Parameters**

format,see	<pre>printf(http://www.cplusplus.com/reference/cstdio/printf/)</pre>
Arbitrary	number of other parameters that where defined in format

Here is the call graph for this function:



6.20.4.12 int AmmServer\_ExecuteCommandLine ( char \* command, char \* what2GetBack, unsigned int what2GetBackMaxSize )

Execute a command and copy its output to the provided buffer.

#### **Parameters**

Command	to execute
Allocated	memory to store the result
Size	of Allocated memory

#### Return values

1=Ok,0=Failed	

**Bug** Executing commands can be dangerous, always check and sanitize input before executing, Also be sure about the max size of output so that you don't lose a part of it, also make something like escapeshellcmd

6.20.4.13 int AmmServer\_FileExists ( char \* filename )

Check if file Exists.

#### **Parameters**

Path	to file
------	---------

#### Return values

1-Eviete 0-Dogs	not Exist
1-LX1313,0-D003	not Exist

Here is the call graph for this function:



6.20.4.14 int AmmServer\_FILES ( struct AmmServer\_Instance \* instance, struct AmmServer\_DynamicRequest \* rqst, char \* var\_id\_IN, char \* var\_value\_OUT, unsigned int max\_var\_value\_OUT)

Access a FILE submitted by a dynamic requested.

#### **Parameters**

Instance	of an AmmarServer
Request	that contains the POST argument ( see AmmServer_DynamicRequest )
Input	Name of argument we are looking for
Output	Pointer that will be copied with the value we were looking for
Maximum	Size for output Value

#### **Return values**

1=Success,0=Failure	

6.20.4.15 int AmmServer\_GETArg ( struct AmmServer\_Instance \* instance, struct AmmServer\_DynamicRequest \* rqst, char \* var\_id\_IN, char \* var\_value\_OUT, unsigned int max\_var\_value\_OUT)

Get a GET argument.

#### **Parameters**

Instance	of an AmmarServer
Request	that contains the POST argument ( see AmmServer_DynamicRequest )
Input	Name of argument we are looking for
Output	Pointer that will be copied with the value we were looking for
Maximum	Size for output Value

#### Return values

1=Success,0=Failure	

Here is the call graph for this function:



### 6.20.4.16 int AmmServer\_GetInfo ( struct AmmServer\_Instance \* instance, unsigned int info\_type )

Get an Integer out of the state of an instance , of course one can dive into the instance structure but this is a much more clean way to do this.

#### **Parameters**

An	AmmarServer Instance
An	ID about which info we want , see ( AmmServInfos )

### Return values

Value	of the integer we asked about

### $6.20.4.17 \quad \text{int AmmServer\_GetIntSettingValue ( struct AmmServer\_Instance} * \textit{instance}, \text{ unsigned int } \textit{set\_type} \text{ )}$

Get an Integer out of the state of an instance , of course one can dive into the instance structure but this is a much more clean way to do this.

### **Parameters**

An	AmmarServer Instance
An	ID about which integer info we want , see ( AmmServSettings )

#### Return values

Value	of the integer we asked about
-------	-------------------------------

### 6.20.4.18 char\* AmmServer\_GetStrSettingValue ( struct AmmServer\_Instance \* instance, unsigned int set\_type )

Get a String out of the state of an instance , of course one can dive into the instance structure but this is a much more clean way to do this.

#### **Parameters**

An	AmmarServer Instance
An	ID about which string info we want , see ( AmmServStrSettings )

#### Return values

Value	of the string we asked about

6.20.4.19 int AmmServer\_POSTArg ( struct AmmServer\_Instance \* instance, struct AmmServer\_DynamicRequest \* rqst, char \* var\_id\_IN, char \* var\_value\_OUT, unsigned int max\_var\_value\_OUT)

### Get a POST argument.

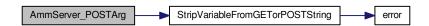
#### **Parameters**

Instance	of an AmmarServer
Request	that contains the POST argument ( see AmmServer_DynamicRequest )
Input	Name of argument we are looking for
Output	Pointer that will be copied with the value we were looking for
Maximum	Size for output Value

#### **Return values**

1=Success,0=Failure	

Here is the call graph for this function:



6.20.4.20 char\* AmmServer\_ReadFileToMemory ( char \* filename, unsigned int \* length )

Read a file and store it to a freshly allocated memory block.

#### **Parameters**

Input	Filename
Output	Maximum Size

#### Return values

Pointer	to the new memory or 0=Failed

Here is the call graph for this function:



6.20.4.21 int AmmServer\_RegisterTerminationSignal ( void \* callback )

Register a function to call a function that gracefully terminates a client when a SIGKILL or the time to stop the server comes.

#### **Parameters**

Pointer	to function
---------	-------------

#### **Return values**

1=Exists,0=Does	not Exist

Here is the call graph for this function:



6.20.4.22 int AmmServer\_RemoveResourceHandler ( struct AmmServer\_Instance \* instance, struct AmmServer\_RH\_Context \* context, unsigned char free\_mem )

Remove a request handler that hanles dynamic requests.

#### **Parameters**

An	AmmarServer Instance
An	AmmServer_RH_Context to be freed
Switch	to control freeing memory or not for this context (typically should be set to 1 except one
	knows what he is trying to do )

### Return values

1=Success,0=Failure	

Here is the call graph for this function:



6.20.4.23 int AmmServer\_ReplaceVarInMemoryFile ( char \* page, unsigned int pageLength, char \* var, char \* value )

Hot-Replace a variable inside a memory block , typically used to replace placeholders inside text files , like \$\$\$\$\$\$NAME\$\$\$\$\$\$, the value should be smaller or equal to the var beeing replaced.

#### **Parameters**

Pointer	to memory that contains the document
Size	of document
Variable	to be replaced

What	to replace it with

Return values

1=Ok,0=Failed	

Bug Value should not be bigger than variable otherwise things won't fit in the same memory block

6.20.4.24 int AmmServer\_Running ( struct AmmServer\_Instance \* instance )

Query if an instance of AmmarServer is initialized and running.

#### **Parameters**

An	AmmarServer Instance

#### **Return values**

```
1=Running,0=Stopped
```

Here is the call graph for this function:



6.20.4.25 int AmmServer\_SaveDynamicRequest ( char \* filename, struct AmmServer\_Instance \* instance, struct AmmServer\_DynamicRequest \* rqst )

Save Dynamic Request to file.

#### **Parameters**

Filename	to save the dynamic request
Instance	of an AmmarServer
Request	that we want to save to a file ( see AmmServer_DynamicRequest )

### Return values

1=Success,0=Failure	

Here is the call graph for this function:



6.20.4.26 int AmmServer\_SelfCheck ( struct AmmServer\_Instance \* instance )

Perform a sanity check on the instance of AmmarServer , this is mostly a dev debug tool and an entry point for code inside AmmServerlib.

#### **Parameters**

Ammar	Server Instance
-------	-----------------

#### Return values

1=Ok,0=Failed	

Bug Maybe remove AmmServer\_SelfCheck

6.20.4.27 int AmmServer\_SetIntSettingValue ( struct AmmServer\_Instance \* instance, unsigned int set\_type, int set\_value )

Set an Integer inside the state of an instance , of course one can dive into the instance structure but this is a much more clean way to do this.

#### **Parameters**

An	AmmarServer Instance
An	ID about which integer info we want , see ( AmmServSettings )
New	value to set

#### **Return values**

Value	of the integer we asked about

6.20.4.28 int AmmServer\_SetStrSettingValue ( struct AmmServer\_Instance \* instance, unsigned int set\_type, char \* set\_value )

Set an string inside the state of an instance , of course one can dive into the instance structure but this is a much more clean way to do this.

### **Parameters**

An	AmmarServer Instance
An	ID about which integer info we want , see ( AmmServStrSettings )
New	string value to set

### Return values

1=Success,0=Failure	

Here is the call graph for this function:



6.20.4.29 int AmmServer\_SignalCountAsBadClientBehaviour ( struct AmmServer\_Instance \* instance, struct AmmServer\_DynamicRequest \* rqst )

Staged way to easily handle bad clients etc from the clients , currently a stub..!

Bug Client behaviours etc are not implemented yet

6.20.4.30 struct AmmServer\_Instance\* AmmServer\_Start ( char \* name, char \* ip, unsigned int port, char \* conf\_file, char \* web\_root\_path, char \* templates\_root\_path )

Start a Web Server , allocate memory , bind ports and return its instance..

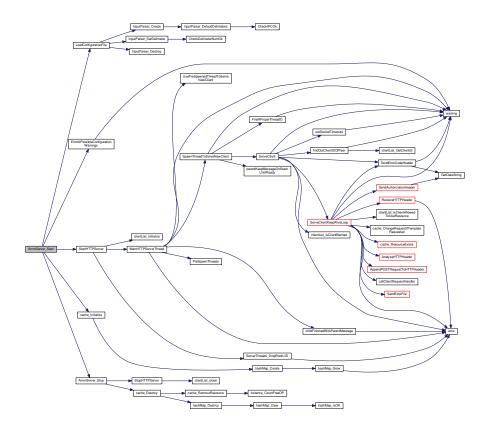
#### **Parameters**

String	containing the name of this Server
String	containing the IP to be binded ( 0.0.0.0 , for all interfaces )
Port	to use , ports under 1000 require superuser privileges
String	with the filename of a configuration file
String	with the root public_html directory, all directories that are childs of this dir could be visible
String	with the root directory for templates ( custom 404 pages etc )

### Return values

An	Ammar Server instance or 0=Failure

Here is the call graph for this function:



6.20.4.31 struct AmmServer\_Instance\* AmmServer\_StartAdminInstance ( char \* ip, unsigned int port )

Planned functionality for a default http administrator panel per server per instance , currently not implemented correctly.

#### **Parameters**

IP	to bind the interface at
Port	to use

#### Return values

Value	of the integer we asked about

6.20.4.32 struct AmmServer\_Instance\* AmmServer\_StartWithArgs ( char \* name, int argc, char \*\* argv, char \* ip, unsigned int port, char \* conf\_file, char \* web\_root\_path, char \* templates\_root\_path )

Start a Web Server , allocate memory , bind ports and return its instance , also process arguments ( argc and argv from int main(int argc, char \*argv[]) ) ..

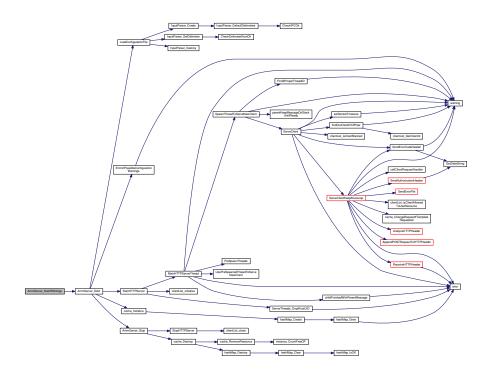
#### **Parameters**

String	containing the name of this Server
argc,number	of arguments
argv,array	of strings
String	containing the IP to be binded ( 0.0.0.0 , for all interfaces )
Port	to use , ports under 1000 require superuser privileges
String	with the filename of a configuration file
String	with the root public_html directory , all directories that are childs of this dir could be visible
String	with the root directory for templates ( custom 404 pages etc )

### Return values

An	Ammar Server instance or 0=Failure

Here is the call graph for this function:



6.20.4.33 int AmmServer\_Stop ( struct AmmServer\_Instance \* instance )

Stop a Web Server , deallocate memory , free ports and free the server instance..

#### **Parameters**

An	AmmarServer Instance
----	----------------------

#### Return values

```
1=Success,0=Failure
```

Here is the call graph for this function:



### 6.20.4.34 unsigned int AmmServer\_StringlsHTMLSafe ( char \* str )

Check if a string has html elements inside it, so if we append it to a web site we won't have html injected.

#### **Parameters**

Innut	String
IIIDUL	i Silliu
1	

#### Return values

## 6.20.4.35 void AmmServer\_Success ( const char \* format, ... )

Writes the C string pointed by format to stderr, as a success ( Green ) and logs it to the appropriate log If format includes format specifiers (subsequences beginning with %), the additional arguments following format are formatted and inserted in the resulting string replacing their respective specifiers.

# Parameters

format,see	<pre>printf(http://www.cplusplus.com/reference/cstdio/printf/)</pre>
Arbitrary	number of other parameters that where defined in format

Here is the call graph for this function:



```
6.20.4.36 char* AmmServer_Version ( )
```

Returns a string with the version of AmmarServer , in case it returns NULL it means that we are linked to AmmarServerNULL which means a fake binary.

```
6.20.4.37 void AmmServer_Warning (const char * format, ...)
```

Writes the C string pointed by format to stderr , as a warning ( Yellow ) and logs it to the appropriate log If format includes format specifiers (subsequences beginning with %), the additional arguments following format are formatted and inserted in the resulting string replacing their respective specifiers.

#### **Parameters**

format,see	<pre>printf(http://www.cplusplus.com/reference/cstdio/printf/)</pre>
Arbitrary	number of other parameters that where defined in format

Here is the call graph for this function:



6.20.4.38 int AmmServer\_WriteFileFromMemory ( char \* filename, char \* memory, unsigned int memoryLength )

Dump a memory block to a file.

### Parameters

Output	Filename
Input	Pointer to memory
Size	of memory block

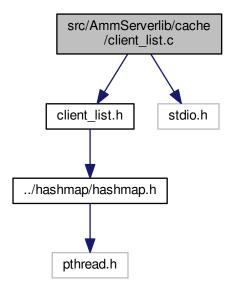
#### **Return values**

1=Ok,0=Failed	

# 6.21 src/AmmServerlib/cache/client\_list.c File Reference

```
#include "client_list.h"
#include <stdio.h>
```

Include dependency graph for client\_list.c:



#### **Macros**

• #define COMPILE\_WITH\_CLIENT\_LIST 1

### **Functions**

- unsigned int clientList\_GetClientId (struct clientListContext \*clientList, char \*ip)
  - Get the internal index id of an IP.
- int clientList\_isClientBanned (struct clientListContext \*clientList, clientID client\_id)
  - Check if client ID is banned, therefore we should deny all service to him.
- int clientList\_isClientAllowedToUseResource (struct clientListContext \*clientList, clientID client\_id, char \*resource)

Ask if the client is allowed to use resource.

 int clientList\_signalClientStoppedUsingResource (struct clientListContext \*clientList, clientID client\_id, char \*resource)

Signal that resource has stopped beeing used for internal statistics.

- struct clientListContext \* clientList\_initialize ()
  - Create, allocate and return a client list.
- int clientList\_close (struct clientListContext \*clientList)

Close and destroy client list.

- 6.21.1 Macro Definition Documentation
- 6.21.1.1 #define COMPILE\_WITH\_CLIENT\_LIST 1
- 6.21.2 Function Documentation

 $\textbf{6.21.2.1} \quad \text{int clientList\_close} \ ( \ \textbf{struct clientListContext} \ * \ \textbf{\textit{clientList}} \ )$ 

Close and destroy client list.

#### **Parameters**

ClientList	to destroy
------------	------------

#### **Return values**

1=Success,0=Failure	

6.21.2.2 unsigned int clientList\_GetClientId ( struct clientListContext \* clientList, char \* ip )

Get the internal index id of an IP.

#### **Parameters**

Client	List	
Str	ing	containing the IP of the client we want to query

#### **Return values**

ID	of client we searched for

6.21.2.3 struct clientListContext\* clientList\_initialize ( )

Create, allocate and return a client list.

#### **Return values**

Pointer	to a freshly allocated client list or 0=Failure

6.21.2.4 int clientList\_isClientAllowedToUseResource ( struct clientListContext \* clientList, clientID client\_id, char \* resource )

Ask if the client is allowed to use resource.

### Parameters

ClientList	
ClientID	we are talking about
String	of the resource

#### Return values

1=Allowed,0=Denied	

6.21.2.5 int clientList\_isClientBanned ( struct clientListContext \* clientList, clientID client\_id )

Check if client ID is banned, therefore we should deny all service to him.

### **Parameters**

ClientList	
ClientID	we are asking about

#### **Return values**

-	Don	d	າ=OK
- 1	=Dan	nea.	<i>リ</i> =し / ハ

6.21.2.6 int clientList\_signalClientStoppedUsingResource ( struct clientListContext \* clientList, clientID client\_id, char \* resource )

Signal that resource has stopped beeing used for internal statistics.

#### **Parameters**

ClientList	
ClientID	we are talking about
String	of the resource

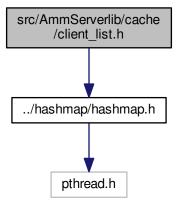
#### Return values

1=Ok,0=Failed	

# 6.22 src/AmmServerlib/cache/client\_list.h File Reference

Client list for IPs that should also serve as a banlist manage QoS etc.

#include "../hashmap/hashmap.h"
Include dependency graph for client\_list.h:



This graph shows which files directly or indirectly include this file:



#### **Data Structures**

· struct clientListContext

The client list is just a hashmap ( see hashmap.h )

### **Typedefs**

typedef unsigned int clientID

Typedef to make clientID stand out.

#### **Functions**

• unsigned int clientList\_GetClientId (struct clientListContext \*clientList, char \*ip)

Get the internal index id of an IP.

int clientList\_isClientBanned (struct clientListContext \*clientList, clientID client\_id)

Check if client ID is banned, therefore we should deny all service to him.

 int clientList\_isClientAllowedToUseResource (struct clientListContext \*clientList, clientID client\_id, char \*resource)

Ask if the client is allowed to use resource.

 int clientList\_signalClientStoppedUsingResource (struct clientListContext \*clientList, clientID client\_id, char \*resource)

Signal that resource has stopped beeing used for internal statistics.

struct clientListContext \* clientList initialize ()

Create, allocate and return a client list.

• int clientList\_close (struct clientListContext \*clientList)

Close and destroy client list.

### 6.22.1 Detailed Description

Client list for IPs that should also serve as a banlist manage QoS etc.

Author

Ammar Qammaz (AmmarkoV)

Bug Client Lists are a stub and not implemented yet

#### 6.22.2 Typedef Documentation

6.22.2.1 typedef unsigned int clientID

Typedef to make clientID stand out.

### 6.22.3 Function Documentation

6.22.3.1 int clientList\_close ( struct clientListContext \* clientList )

Close and destroy client list.

#### **Parameters**

ClientList	to destroy
------------	------------

#### Return values

```
1=Success,0=Failure
```

6.22.3.2 unsigned int clientList\_GetClientId ( struct clientListContext \* clientList, char \* ip )

Get the internal index id of an IP.

#### **Parameters**

ClientList	
String	containing the IP of the client we want to query

#### Return values

ID	of client we searched for

6.22.3.3 struct clientListContext\* clientList\_initialize ( )

Create, allocate and return a client list.

#### **Return values**

Pointer	to a freshly allocated client list or 0=Failure

6.22.3.4 int clientList\_isClientAllowedToUseResource ( struct clientListContext \* clientList, clientID client\_id, char \* resource )

Ask if the client is allowed to use resource.

### Parameters

ClientList	
ClientID	we are talking about
String	of the resource

#### Return values

1=Allowed,0=Denied	

6.22.3.5 int clientList\_isClientBanned ( struct clientListContext \* clientList, clientID client\_id )

Check if client ID is banned, therefore we should deny all service to him.

### **Parameters**

ClientList	
ClientID	we are asking about

### Return values

Generated on Thu Jul 17 2014 15:44:54 for AmmarServer by Doxygen

1_	Ва	nn	00	1	)_/	$\cap \iota$	1
1=	Da.	IIII	ec	ı.u	=	Jr	١

6.22.3.6 int clientList\_signalClientStoppedUsingResource ( struct clientListContext \* clientList, clientID client\_id, char \* resource )

Signal that resource has stopped beeing used for internal statistics.

#### **Parameters**

ClientList	
ClientID	we are talking about
String	of the resource

#### Return values

```
1=Ok,0=Failed
```

# 6.23 src/AmmServerlib/cache/dynamic\_requests.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "dynamic_requests.h"
#include "file_caching.h"
#include "../server_configuration.h"
#include "../tools/logs.h"
#include "../tools/time_provider.h"
Include dependency graph for dynamic requests.c:
```

stdio.h stdib.h string.h file\_caching.h .../server\_configuration.h .../tools/logs.h .../tools/time\_provider.h .../server\_configuration.h .../tools/logs.h .../tools/time\_provider.h .../server\_configuration.h .../tools/logs.h .../tools/time\_provider.h .../tools/http\_tools.h .../server\_configuration.h .../server\_configuration.h .../tools/logs.h .../tools/time\_provider.h .../server\_configuration.h .../server\_configuration.h .../server\_configuration.h .../tools/logs.h .../tools/time\_provider.h .../server\_configuration.h .../tools/logs.h .../tools/time\_provider.h .../server\_configuration.h .../server\_configuration.h .../server\_configuration.h .../tools/logs.h .../tools/time\_provider.h .../server\_configuration.h .../serv

### **Functions**

- int dynamicRequest\_ContentAvailiable (struct AmmServer\_Instance \*instance, unsigned int index)

  Ask if dynamic content is available for this cache index.
- char \* dynamicRequest\_serveContent (struct AmmServer\_Instance \*instance, struct HTTPHeader \*request, struct AmmServer\_RH\_Context \*shared\_context, unsigned int index, unsigned long \*memSize, unsigned char \*compressionSupported, unsigned char \*freeContentAfterUsingIt)

Handles and serves a dynamic request.

- int callClientRequestHandler (struct AmmServer\_Instance \*instance, struct HTTPHeader \*output)

  Execute callback function associated with dynamic content, providing it with the http header it needs to output data to
- int saveDynamicRequest (char \*filename, struct AmmServer\_Instance \*instance, struct AmmServer\_DynamicRequest \*rqst)

Save Dynamic request to a file ( for debugging it )

#### 6.23.1 Function Documentation

6.23.1.1 int callClientRequestHandler ( struct AmmServer\_Instance \* instance, struct HTTPHeader \* output )

Execute callback function associated with dynamic content , providing it with the http header it needs to output data to.

#### **Parameters**

An	AmmarServer Instance
HTTPHeader	containing the output of the request

#### Return values

1=Ok,0=Failed	

6.23.1.2 int dynamicRequest\_ContentAvailiable ( struct AmmServer\_Instance \* instance, unsigned int index )

Ask if dynamic content is available for this cache index.

#### **Parameters**

An	AmmarServer Instance
Index	of cache we want to ask for

### Return values

1=Availiable,0=Not	Availiable

6.23.1.3 char\* dynamicRequest\_serveContent ( struct AmmServer\_Instance \* instance, struct HTTPHeader \* request, struct AmmServer\_RH\_Context \* shared\_context, unsigned int index, unsigned long \* memSize, unsigned char \* compressionSupported, unsigned char \* freeContentAfterUsingIt )

Handles and serves a dynamic request.

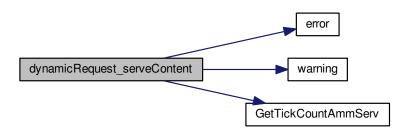
### **Parameters**

An	AmmarServer Instance
HTTPHeader	containing the request done
Resource	Context for specific dynamic Request
Index	of cache item , containing this dynamic request
Memory	Size allocated by the new dynamic request
Outputs	if compression was supported ( and used ) by client
Outputs	if client wants to free buffer on it's own or it should be handled automatically

#### **Return values**

Pointer	To New Content or ,0=Failed

Here is the call graph for this function:



6.23.1.4 int saveDynamicRequest ( char \* filename, struct AmmServer\_Instance \* instance, struct AmmServer\_DynamicRequest \* rqst )

Save Dynamic request to a file ( for debugging it )

### **Parameters**

ClientList	
ClientID	we are talking about
String	of the resource

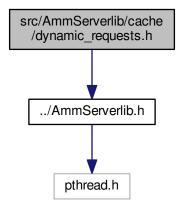
### Return values

1=Ok,0=Failed	

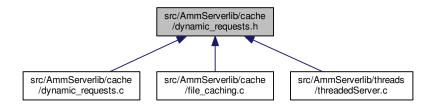
# 6.24 src/AmmServerlib/cache/dynamic\_requests.h File Reference

Dynamic request handler , one of the most important parts of this library.

#include "../AmmServerlib.h"
Include dependency graph for dynamic\_requests.h:



This graph shows which files directly or indirectly include this file:



### **Functions**

- int dynamicRequest\_ContentAvailiable (struct AmmServer\_Instance \*instance, unsigned int index)

  Ask if dynamic content is available for this cache index.
- char \* dynamicRequest\_serveContent (struct AmmServer\_Instance \*instance, struct HTTPHeader \*request, struct AmmServer\_RH\_Context \*shared\_context, unsigned int index, unsigned long \*memSize, unsigned char \*compressionSupported, unsigned char \*freeContentAfterUsingIt)

Handles and serves a dynamic request.

- int callClientRequestHandler (struct AmmServer\_Instance \*instance, struct HTTPHeader \*output)
  - Execute callback function associated with dynamic content, providing it with the http header it needs to output data to.
- int saveDynamicRequest (char \*filename, struct AmmServer\_Instance \*instance, struct AmmServer\_DynamicRequest \*rqst)

Save Dynamic request to a file ( for debugging it )

### 6.24.1 Detailed Description

Dynamic request handler, one of the most important parts of this library.

Author

Ammar Qammaz (AmmarkoV)

**Bug** Compression should be improved

#### 6.24.2 Function Documentation

6.24.2.1 int callClientRequestHandler ( struct AmmServer\_Instance \* instance, struct HTTPHeader \* output )

Execute callback function associated with dynamic content , providing it with the http header it needs to output data to

#### **Parameters**

An	AmmarServer Instance
HTTPHeader	containing the output of the request

#### Return values

1=Ok,0=Failed	

6.24.2.2 int dynamicRequest\_ContentAvailiable ( struct AmmServer\_Instance \* instance, unsigned int index )

Ask if dynamic content is available for this cache index.

#### **Parameters**

An	AmmarServer Instance
Index	of cache we want to ask for

### Return values

1=Availiable,0=Not	Availiable

6.24.2.3 char\* dynamicRequest\_serveContent ( struct AmmServer\_Instance \* instance, struct HTTPHeader \* request, struct AmmServer\_RH\_Context \* shared\_context, unsigned int index, unsigned long \* memSize, unsigned char \* compressionSupported, unsigned char \* freeContentAfterUsingIt )

Handles and serves a dynamic request.

#### **Parameters**

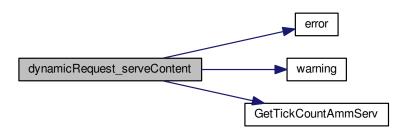
An	AmmarServer Instance
HTTPHeader	containing the request done
Resource	Context for specific dynamic Request
Index	of cache item , containing this dynamic request
Memory	Size allocated by the new dynamic request
Outputs	if compression was supported ( and used ) by client

Outputs	if client wants to free buffer on it's own or it should be handled automatically
D. I.	

Return values

Pointer	To New Content or ,0=Failed
---------	-----------------------------

Here is the call graph for this function:



# 6.24.2.4 int saveDynamicRequest ( char \* filename, struct AmmServer\_Instance \* instance, struct AmmServer\_DynamicRequest \* rqst )

Save Dynamic request to a file (for debugging it)

#### **Parameters**

ClientList	
ClientID	we are talking about
String	of the resource

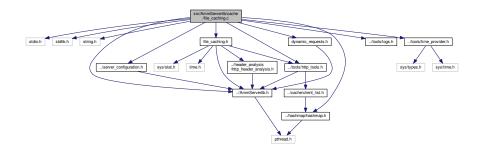
### Return values

```
1=Ok,0=Failed
```

# 6.25 src/AmmServerlib/cache/file\_caching.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "../AmmServerlib.h"
#include "../server_configuration.h"
#include "file_caching.h"
#include "../tools/logs.h"
#include "../tools/http_tools.h"
#include "../tools/time_provider.h"
#include "../hashmap/hashmap.h"
#include "dynamic_requests.h"
```

Include dependency graph for file\_caching.c:



#### **Functions**

 int cache\_ChangeRequestIfTemplateRequested (struct AmmServer\_Instance \*instance, char \*request, char \*templates\_root)

The role of request caching is to intercept incoming requests and if they are referring to an internal resource using the TemplatesInternalURI URI we want to redirect the request to our templates folder ..! If the request was indeed a change request returns 1 else 0.

• int freeMallocIfNeeded (char \*mem, unsigned char free is needed)

Tool to check if a malloc'ed chunk of memory should be freed.

• int cache\_Initialize (struct AmmServer\_Instance \*instance, unsigned int max\_seperate\_items, unsigned int max\_total\_allocation\_MB, unsigned int max\_allocation\_per\_entry\_MB)

Allocate and create a new empty cache.

int cache\_Destroy (struct AmmServer\_Instance \*instance)

Deallocate and destroy the cache of an AmmarServer instance.

unsigned int cache\_FindResource (struct AmmServer\_Instance \*instance, char \*resource, unsigned int \*index)

Query for a resource, and return its index.

- int cache\_CreateResource (struct AmmServer\_Instance \*instance, char \*resource, unsigned int \*index)
- int cache DestroyResource (unsigned int \*index)
- int cache\_LoadResourceFromDisk (struct AmmServer\_Instance \*instance, char \*filename, unsigned int \*index)
- int cache\_AddFile (struct AmmServer\_Instance \*instance, char \*filename, unsigned int \*index, struct stat \*last modification)

Add a filesystem file to cache.

int cache\_AddMemoryBlock (struct AmmServer\_Instance \*instance, struct AmmServer\_RH\_Context \*context)

Add a memory block to cache.

• int cache\_AddDoNOTCacheRuleForResource (struct AmmServer\_Instance \*instance, char \*filename)

Create a rule for specific resource so that it will always be served fresh and not cached.

- int cache\_RemoveResource (struct AmmServer\_Instance \*instance, unsigned int index)
- int cache\_RemoveContextAndResource (struct AmmServer\_Instance \*instance, struct AmmServer\_RH\_-Context \*context, unsigned char free\_mem)

Destroy Cache entry and resource context.

• unsigned long cache\_GetHashOfResource (struct AmmServer\_Instance \*instance, unsigned int index)

Get Hash Value of resource ( to be used for E-Tags http://en.wikipedia.org/wiki/HTTP\_ETag or whatever other reason)

• int cache\_ResourceExists (struct AmmServer\_Instance \*instance, char \*verified\_filename, unsigned int \*index)

Query for a resource, and return its index.

char \* cache\_GetResource (struct AmmServer\_Instance \*instance, struct HTTPHeader \*request, unsigned int resourceCacheID, char \*verified\_filename, unsigned int \*index, unsigned long \*filesize, struct stat \*last\_modification, unsigned char \*compressionSupported, unsigned char \*freeContentAfterUsingIt)

Get a resource to be served to a client . This call will try to find if it is already used , if it exists on disk , if it is a dynamic request etc , and return the specified buffer.

### 6.25.1 Function Documentation

6.25.1.1 int cache\_AddDoNOTCacheRuleForResource ( struct AmmServer Instance \* instance, char \* filename )

Create a rule for specific resource so that it will always be served fresh and not cached.

### **Parameters**

An	AmmarServer Instance	
Resource	filename	

#### **Return values**

1=Success,0=Failure	

Here is the call graph for this function:



6.25.1.2 int cache\_AddFile ( struct AmmServer\_Instance \* instance, char \* filename, unsigned int \* index, struct stat \* last\_modification )

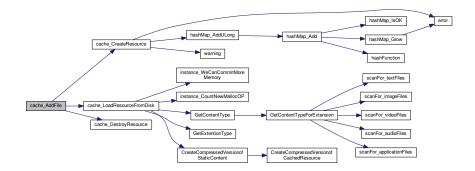
Add a filesystem file to cache.

#### **Parameters**

An	AmmarServer Instance
Filename	pointing to the file to be added to cache
Output	index number of cache item
Output	time of last modification

1=Found,0=Failed	
------------------	--

Here is the call graph for this function:



6.25.1.3 int cache\_AddMemoryBlock ( struct AmmServer\_Instance \* instance, struct AmmServer\_RH\_Context \* context )

Add a memory block to cache.

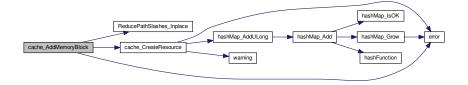
### **Parameters**

An	AmmarServer Instance
Dynamic	Request to be added

### **Return values**

```
1=Success,0=Failure
```

Here is the call graph for this function:



6.25.1.4 int cache\_ChangeRequestlfTemplateRequested ( struct AmmServer\_Instance \* instance, char \* request, char \* templates\_root )

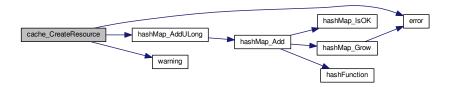
The role of request caching is to intercept incoming requests and if they are referring to an internal resource using the TemplatesInternalURI URI we want to redirect the request to our templates folder ..! If the request was indeed a change request returns 1 else 0.

An	AmmarServer Instance	
String	of Request	

Filename	pointing to	directory that contains templates	
Return values			
	1_lf	request was for a template and it got changed $\Omega$ not changed request	

6.25.1.5 int cache\_CreateResource ( struct AmmServer\_Instance \* instance, char \* resource, unsigned int \* index )

Here is the call graph for this function:



6.25.1.6 int cache\_Destroy ( struct AmmServer\_Instance \* instance )

Deallocate and destroy the cache of an AmmarServer instance.

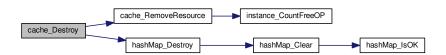
### **Parameters**

An	AmmarServer Instance
	Animaiserver instance

## Return values

```
1=Ok,0=Failed
```

Here is the call graph for this function:



- 6.25.1.7 int cache\_DestroyResource ( unsigned int \* index )
- 6.25.1.8 unsigned int cache\_FindResource ( struct AmmServer\_Instance \* instance, char \* resource, unsigned int \* index )

Query for a resource, and return its index.

An	AmmarServer Instance
Resource	we are searching for
Output	Index number

### **Return values**

1=Success,0=Failure	

Here is the call graph for this function:



6.25.1.9 unsigned long cache\_GetHashOfResource ( struct AmmServer\_Instance \* instance, unsigned int index )

Get Hash Value of resource ( to be used for E-Tags  $http://en.wikipedia.org/wiki/HTTP\_ETag$  or whatever other reason )

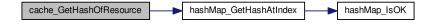
### **Parameters**

An	AmmarServer Instance
Index	to the cache item requested

### Return values

Hash	Value for Index specified , 0 = failure

Here is the call graph for this function:



6.25.1.10 char\* cache\_GetResource ( struct AmmServer\_Instance \* instance, struct HTTPHeader \* request, unsigned int resourceCacheID, char \* verified\_filename, unsigned int \* index, unsigned long \* filesize, struct stat \* last\_modification, unsigned char \* compressionSupported, unsigned char \* freeContentAfterUsingIt )

Get a resource to be served to a client . This call will try to find if it is already used , if it exists on disk , if it is a dynamic request etc , and return the specified buffer.

An	AmmarServer Instance

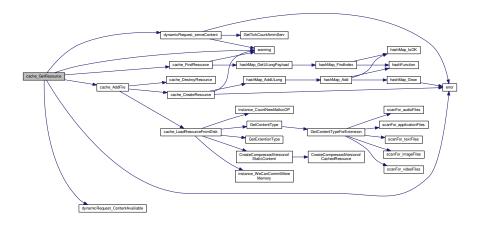
HTTPHeader	of request we are trying to service with the resource
cacheID	for resource
Filename	of the resource, this should be verified so that it doesn't access the whole filesystem but only
	subdirectories of the root public_html dir , and we consider this safe
Output	Index number of cache item we requested
Output	FileSize of cache item we requested
Output	last modification time of cache item we requested
Output	flag about whether the buffer returned is compressed or not
Output	flag about whether it is safe to automatically free the resource after using it, or there is an
	automatic handling of memory for the specific item

**Bug** If verified\_filename, is not really verified (i.e. outside of the public\_html root directory, this function could pose a security problem, since it will just blindly open and serve the filename given to it)

#### **Return values**

1=Ok,0=Failed	

Here is the call graph for this function:



6.25.1.11 int cache\_Initialize ( struct AmmServer\_Instance \* instance, unsigned int max\_seperate\_items, unsigned int max\_total\_allocation\_MB, unsigned int max\_allocation\_per\_entry\_MB )

Allocate and create a new empty cache.

### **Parameters**

An	n AmmarServer Instance	
Maximum	Number of separate items	
Maximum	memory usage ( Megabytes ) for this entire cache	
Maximum	memory usage ( Megabytes ) for a specific entry of the cache	

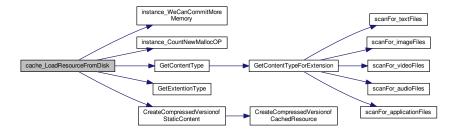
1=Ok,0=Failed	
---------------	--

Here is the call graph for this function:



6.25.1.12 int cache\_LoadResourceFromDisk ( struct AmmServer\_Instance \* instance, char \* filename, unsigned int \* index )

Here is the call graph for this function:



6.25.1.13 int cache\_RemoveContextAndResource ( struct AmmServer\_Instance \* instance, struct AmmServer\_RH\_Context \* context, unsigned char free\_mem )

Destroy Cache entry and resource context.

#### **Parameters**

An	AmmarServer Instance
Resource	Context to be removed
Flag	controlling whether memory should be freed

## Return values

1=Success,0=Failure
---------------------

Here is the call graph for this function:



6.25.1.14 int cache\_RemoveResource ( struct AmmServer\_Instance \* instance, unsigned int index )

Here is the call graph for this function:



6.25.1.15 int cache\_ResourceExists ( struct AmmServer\_Instance \* instance, char \* verified\_filename, unsigned int \* index )

Query for a resource, and return its index.

### **Parameters**

An	AmmarServer Instance
Resource	we are searching for
Output	Index number

#### Return values

1=Success,0=Failure	

Here is the call graph for this function:



6.25.1.16 int freeMalloclfNeeded ( char \* mem, unsigned char free\_is\_needed )

Tool to check if a malloc'ed chunk of memory should be freed.

### **Parameters**

Pointer	to memory
Flag	that signals if the pointer should be freed or not

## Return values

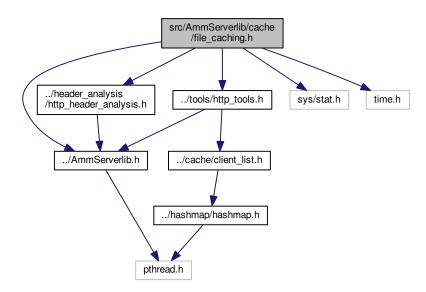
1=Ok,0=Failed		
	1=Ok.0=Failed	

# 6.26 src/AmmServerlib/cache/file\_caching.h File Reference

Central cache of AmmarServer , it reads/indexes and swaps resources asked by clients for fast performance.

```
#include "../AmmServerlib.h"
#include "../header_analysis/http_header_analysis.h"
#include "../tools/http_tools.h"
#include <sys/stat.h>
#include <time.h>
```

Include dependency graph for file caching.h:



This graph shows which files directly or indirectly include this file:



## **Data Structures**

struct timestamp

Timestamp for a cache item entry.

struct cache\_item

A cache item and all it's contents.

### **Functions**

• int freeMallocIfNeeded (char \*mem, unsigned char free\_is\_needed)

Tool to check if a malloc'ed chunk of memory should be freed.

• int cache\_ChangeRequestIfTemplateRequested (struct AmmServer\_Instance \*instance, char \*request, char \*templates\_root)

The role of request caching is to intercept incoming requests and if they are referring to an internal resource using the TemplatesInternalURI URI we want to redirect the request to our templates folder ..! If the request was indeed a change request returns 1 else 0.

• int cache\_AddFile (struct AmmServer\_Instance \*instance, char \*filename, unsigned int \*index, struct stat \*last modification)

Add a filesystem file to cache.

int cache\_AddMemoryBlock (struct AmmServer\_Instance \*instance, struct AmmServer\_RH\_Context \*context)

Add a memory block to cache.

- int cache\_AddDoNOTCacheRuleForResource (struct AmmServer\_Instance \*instance, char \*filename)
  - Create a rule for specific resource so that it will always be served fresh and not cached.
- int cache\_RemoveContextAndResource (struct AmmServer\_Instance \*instance, struct AmmServer\_RH\_-Context \*context, unsigned char free\_mem)

Destroy Cache entry and resource context.

- unsigned long cache\_GetHashOfResource (struct AmmServer\_Instance \*instance, unsigned int index)
  - Get Hash Value of resource ( to be used for E-Tags http://en.wikipedia.org/wiki/HTTP\_ETag or whatever other reason)
- unsigned int cache\_FindResource (struct AmmServer\_Instance \*instance, char \*resource, unsigned int \*index)

Query for a resource, and return its index.

• int cache\_ResourceExists (struct AmmServer\_Instance \*instance, char \*verified\_filename, unsigned int \*index)

Query for a resource, and return its index.

• int cache\_Initialize (struct AmmServer\_Instance \*instance, unsigned int max\_seperate\_items, unsigned int max\_total\_allocation\_MB, unsigned int max\_allocation\_per\_entry\_MB)

Allocate and create a new empty cache.

- int cache\_Destroy (struct AmmServer\_Instance \*instance)
  - Deallocate and destroy the cache of an AmmarServer instance.
- char \* cache\_GetResource (struct AmmServer\_Instance \*instance, struct HTTPHeader \*request, unsigned int resourceCacheID, char \*verified\_filename, unsigned int \*index, unsigned long \*filesize, struct stat \*last\_modification, unsigned char \*compressionSupported, unsigned char \*freeContentAfterUsingIt)

Get a resource to be served to a client . This call will try to find if it is already used , if it exists on disk , if it is a dynamic request etc , and return the specified buffer.

### 6.26.1 Detailed Description

Central cache of AmmarServer, it reads/indexes and swaps resources asked by clients for fast performance.

Author

Ammar Qammaz (AmmarkoV)

**Bug** File caching relies on hashmap for storing data, so it relies on optimizations done there for seek time optimization, other than that there needs to be a clean-up and code quality improvement

### 6.26.2 Function Documentation

6.26.2.1 int cache AddDoNOTCacheRuleForResource ( struct AmmServer Instance \* instance, char \* filename )

Create a rule for specific resource so that it will always be served fresh and not cached.

An	AmmarServer Instance
Resource	filename

## Return values

1=Success,0=Failure	

Here is the call graph for this function:



6.26.2.2 int cache\_AddFile ( struct AmmServer\_Instance \* instance, char \* filename, unsigned int \* index, struct stat \* last\_modification )

Add a filesystem file to cache.

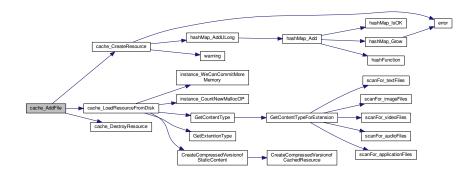
#### **Parameters**

An	AmmarServer Instance
Filename	pointing to the file to be added to cache
Output	index number of cache item
Output	time of last modification

### Return values

1=Found,0=Failed	

Here is the call graph for this function:



6.26.2.3 int cache\_AddMemoryBlock ( struct AmmServer\_Instance \* instance, struct AmmServer\_RH\_Context \* context )

Add a memory block to cache.

#### **Parameters**

An	AmmarServer Instance
Dynamic	Request to be added

#### Return values



Here is the call graph for this function:



6.26.2.4 int cache\_ChangeRequestIfTemplateRequested ( struct AmmServer\_Instance \* instance, char \* request, char \* templates\_root )

The role of request caching is to intercept incoming requests and if they are referring to an internal resource using the TemplatesInternalURI URI we want to redirect the request to our templates folder ..! If the request was indeed a change request returns 1 else 0.

### **Parameters**

An	AmmarServer Instance
String	of Request
Filename	pointing to directory that contains templates

### **Return values**

1=If	request was for a template and it got changed ,0= not changed request

6.26.2.5 int cache\_Destroy ( struct AmmServer\_Instance \* instance )

Deallocate and destroy the cache of an AmmarServer instance.

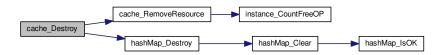
## **Parameters**

An	AmmarServer Instance
----	----------------------

### **Return values**



Here is the call graph for this function:



6.26.2.6 unsigned int cache\_FindResource ( struct AmmServer\_Instance \* instance, char \* resource, unsigned int \* index )

Query for a resource, and return its index.

#### **Parameters**

An	AmmarServer Instance
Resource	we are searching for
Output	Index number

#### Return values

```
1=Success,0=Failure
```

Here is the call graph for this function:



6.26.2.7 unsigned long cache\_GetHashOfResource ( struct AmmServer\_Instance \* instance, unsigned int index )

Get Hash Value of resource ( to be used for E-Tags  $http://en.wikipedia.org/wiki/HTTP\_ETag$  or whatever other reason )

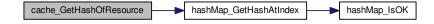
### **Parameters**

An	AmmarServer Instance
Index	to the cache item requested

#### **Return values**

Hash	Value for Index specified , 0 = failure

Here is the call graph for this function:



6.26.2.8 char\* cache\_GetResource ( struct AmmServer\_Instance \* instance, struct HTTPHeader \* request, unsigned int resourceCachelD, char \* verified\_filename, unsigned int \* index, unsigned long \* filesize, struct stat \* last\_modification, unsigned char \* compressionSupported, unsigned char \* freeContentAfterUsingIt )

Get a resource to be served to a client . This call will try to find if it is already used , if it exists on disk , if it is a dynamic request etc , and return the specified buffer.

#### **Parameters**

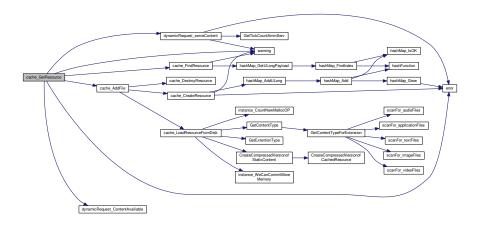
An	AmmarServer Instance
HTTPHeader	of request we are trying to service with the resource
cacheID	for resource
Filename	of the resource, this should be verified so that it doesn't access the whole filesystem but only
	subdirectories of the root public_html dir , and we consider this safe
Output	Index number of cache item we requested
Output	FileSize of cache item we requested
Output	last modification time of cache item we requested
Output	flag about whether the buffer returned is compressed or not
Output	flag about whether it is safe to automatically free the resource after using it, or there is an
	automatic handling of memory for the specific item

**Bug** If verified\_filename, is not really verified (i.e. outside of the public\_html root directory, this function could pose a security problem, since it will just blindly open and serve the filename given to it)

# Return values

1=Ok,0=Failed	
r-On,u-raneu	

Here is the call graph for this function:



6.26.2.9 int cache\_Initialize ( struct AmmServer\_Instance \* instance, unsigned int max\_seperate\_items, unsigned int max\_allocation\_max\_allocation\_per\_entry\_MB )

Allocate and create a new empty cache.

### **Parameters**

An	AmmarServer Instance
Maximum	Number of separate items
Maximum	memory usage ( Megabytes ) for this entire cache
Maximum	memory usage ( Megabytes ) for a specific entry of the cache

1=Ok,0=Failed
---------------

Here is the call graph for this function:



6.26.2.10 int cache\_RemoveContextAndResource ( struct AmmServer\_Instance \* instance, struct AmmServer\_RH\_Context \* context, unsigned char free\_mem )

Destroy Cache entry and resource context.

### **Parameters**

An	AmmarServer Instance
Resource	Context to be removed
Flag	controlling whether memory should be freed

#### Return values

```
1=Success,0=Failure
```

Here is the call graph for this function:



6.26.2.11 int cache\_ResourceExists ( struct AmmServer\_Instance \* instance, char \* verified\_filename, unsigned int \* index )

Query for a resource, and return its index.

#### **Parameters**

An	AmmarServer Instance
Resource	we are searching for
Output	Index number

### Return values



Here is the call graph for this function:



6.26.2.12 int freeMalloclfNeeded ( char \* mem, unsigned char free\_is\_needed )

Tool to check if a malloc'ed chunk of memory should be freed.

#### **Parameters**

Pointer	to memory
Flag	that signals if the pointer should be freed or not

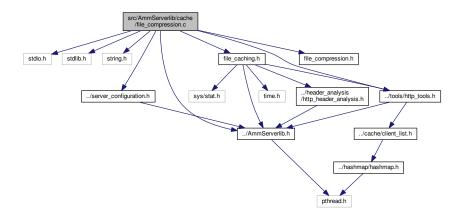
#### Return values

```
1=Ok,0=Failed
```

# 6.27 src/AmmServerlib/cache/file\_compression.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "../AmmServerlib.h"
#include "../server_configuration.h"
#include "../tools/http_tools.h"
#include "file_caching.h"
#include "file_compression.h"
```

Include dependency graph for file compression.c:



## **Functions**

- int CreateCompressedVersionofCachedResource (struct AmmServer\_Instance \*instance, unsigned int index, int compression level)
- int CreateCompressedVersionofDynamicContent (struct AmmServer\_Instance \*instance, unsigned int index)

  Create compressed version of dynamic content, cache item.
- int CreateCompressedVersionofStaticContent (struct AmmServer\_Instance \*instance, unsigned int index)

  Create compressed version of static content, cache item.
- int CreateCompressedVersionofStaticContentPreloading (struct AmmServer\_Instance \*instance, unsigned int index)

Create compressed version of static content which is preloaded, cache item.

## 6.27.1 Function Documentation

6.27.1.1 int CreateCompressedVersionofCachedResource ( struct AmmServer\_Instance \* instance, unsigned int index, int compression\_level ) [inline]

6.27.1.2 int CreateCompressedVersionofDynamicContent ( struct AmmServer Instance \* instance, unsigned int index )

Create compressed version of dynamic content, cache item.

#### **Parameters**

An	AmmarServer Instance
Index	of cache item

#### Return values

```
1=Success,0=Failure
```

Here is the call graph for this function:



6.27.1.3 int CreateCompressedVersionofStaticContent ( struct AmmServer\_Instance \* instance, unsigned int index )

Create compressed version of static content, cache item.

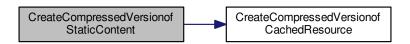
### **Parameters**

An	AmmarServer Instance
Index	of cache item

### **Return values**

1=Success,0=Failure	

Here is the call graph for this function:



6.27.1.4 int CreateCompressedVersionofStaticContentPreloading ( struct AmmServer\_Instance \* instance, unsigned int index )

Create compressed version of static content which is preloaded , cache item.

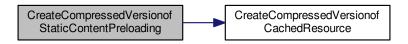
#### **Parameters**

An	AmmarServer Instance
Index	of cache item

#### Return values

1=Success,0=Failure	

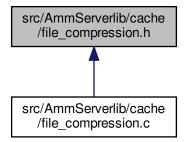
Here is the call graph for this function:



# 6.28 src/AmmServerlib/cache/file\_compression.h File Reference

A tool that compresses memory blocks for better bandwidth usage on the expense of computing power.

This graph shows which files directly or indirectly include this file:



### **Functions**

- int CreateCompressedVersionofDynamicContent (struct AmmServer\_Instance \*instance, unsigned int index)

  Create compressed version of dynamic content, cache item.
- int CreateCompressedVersionofStaticContent (struct AmmServer\_Instance \*instance, unsigned int index)

  Create compressed version of static content, cache item.
- int CreateCompressedVersionofStaticContentPreloading (struct AmmServer\_Instance \*instance, unsigned int index)

Create compressed version of static content which is preloaded, cache item.

## 6.28.1 Detailed Description

A tool that compresses memory blocks for better bandwidth usage on the expense of computing power.

**Author** 

Ammar Qammaz (AmmarkoV)

**Bug** Compression should be improved

## 6.28.2 Function Documentation

6.28.2.1 int CreateCompressedVersionofDynamicContent ( struct AmmServer\_Instance \* instance, unsigned int index )

Create compressed version of dynamic content, cache item.

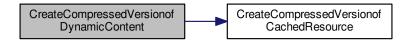
## **Parameters**

An	AmmarServer Instance
Index	of cache item

#### Return values

1=Success,0=Failure	]

Here is the call graph for this function:



6.28.2.2 int CreateCompressedVersionofStaticContent ( struct AmmServer\_Instance \* instance, unsigned int index )

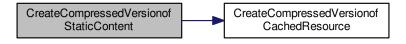
Create compressed version of static content, cache item.

## **Parameters**

An	AmmarServer Instance
Index	of cache item

1=Success,0=Failure
---------------------

Here is the call graph for this function:



6.28.2.3 int CreateCompressedVersionofStaticContentPreloading ( struct AmmServer\_Instance \* instance, unsigned int index )

Create compressed version of static content which is preloaded, cache item.

### **Parameters**

An	AmmarServer Instance
Index	of cache item

### **Return values**

```
1=Success,0=Failure
```

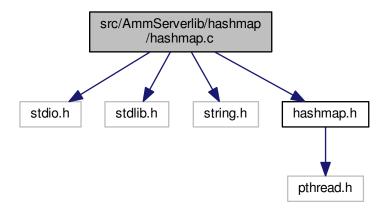
Here is the call graph for this function:



# 6.29 src/AmmServerlib/hashmap/hashmap.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "hashmap.h"
```

Include dependency graph for hashmap.c:



#### **Functions**

unsigned long hashFunction (char \*str)

The function that converts a string to a number so that it will be easier to be searched.

- int hashMap Grow (struct hashMap \*hm, unsigned int growthSize)
- struct hashMap \* hashMap\_Create (unsigned int initialEntries, unsigned int entryAllocationStep, void \*clear-ItemFunction)

Create and allocate a hash map.

- int hashMap\_IsOK (struct hashMap \*hm)
- int hashMap\_GetCurrentNumberOfEntries (struct hashMap \*hm)

Get the current number of entries of hash map.

int hashMap\_GetMaxNumberOfEntries (struct hashMap \*hm)

Get the maximum number of entries of hash map.

- int hashMap IsSorted (struct hashMap \*hm)
- void hashMap Clear (struct hashMap \*hm)

Clear all entries of hash map.

void hashMap\_Destroy (struct hashMap \*hm)

Destroy and deallocate a hash map.

- int cmpHashTableItems (const void \*a, const void \*b)
- int hashMap\_Sort (struct hashMap \*hm)

Sort hash map.

• int hashMap Add (struct hashMap \*hm, char \*key, void \*val, unsigned int valLength)

Add a new key to hash map.

• int hashMap\_AddULong (struct hashMap \*hm, char \*key, unsigned long val)

Add a new key (integer) to hash map.

• int hashMap FindIndex (struct hashMap \*hm, char \*key, unsigned long \*index)

Find index of a key.

int hashmap SwapRecords (struct hashMap \*hm, unsigned int index1, unsigned int index2)

Swap two records.

char \* hashMap GetKeyAtIndex (struct hashMap \*hm, unsigned int index)

Return key value for index.

unsigned long hashMap\_GetHashAtIndex (struct hashMap \*hm, unsigned int index)

Return key hash for index.

int hashMap\_GetPayload (struct hashMap \*hm, char \*key, void \*payload)

Return payload for specified key.

• int hashMap\_GetULongPayload (struct hashMap \*hm, char \*key, unsigned long \*payload)

Return numerical payload for specified key.

int hashMap\_ContainsKey (struct hashMap \*hm, char \*key)

Check if hashmap contains a key.

int hashMap\_ContainsValue (struct hashMap \*hm, void \*val)

Check if hashmap contains a value.

int hashMap\_SaveToFile (struct hashMap \*hm, char \*filename)

Save hash map to a file.

int hashMap\_LoadToFile (struct hashMap \*hm, char \*filename)

Load hash map from a file.

### 6.29.1 Function Documentation

6.29.1.1 int cmpHashTableItems ( const void \*a, const void \*b )

6.29.1.2 unsigned long hashFunction ( char \* str )

The function that converts a string to a number so that it will be easier to be searched.

djb2 This algorithm (k=33) was first reported by dan bernstein many years ago in comp.lang.c. another version of this algorithm (now favored by bernstein) uses xor: hash(i) = hash(i - 1) \* 33  $^{\wedge}$  str[i]; the magic of number 33 (why it works better than many other constants, prime or not) has never been adequately explained. Needless to say , this is our hash function..!

6.29.1.3 int hashMap\_Add ( struct hashMap \* hm, char \* key, void \* val, unsigned int valLength )

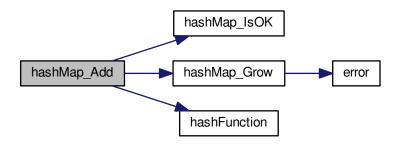
Add a new key to hash map.

#### **Parameters**

HashMap	
String	with the key index
String	with the value of this record
Length	of the value

1=Success,0=Failure
---------------------

Here is the call graph for this function:



6.29.1.4 int hashMap\_AddULong ( struct hashMap \* hm, char \* key, unsigned long val )

Add a new key (integer) to hash map.

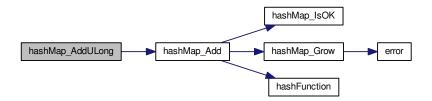
### **Parameters**

HashMap	
String	with the key index
Number	value of this record

## Return values

1=Success,0=Failure	
---------------------	--

Here is the call graph for this function:



6.29.1.5 void hashMap\_Clear ( struct hashMap \*hm )

Clear all entries of hash map.

HashMap	
Return values	

Here is the call graph for this function:

No

return value



6.29.1.6 int hashMap\_ContainsKey ( struct hashMap \* hm, char \* key )

Check if hashmap contains a key.

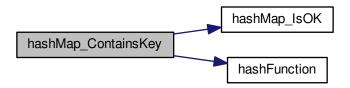
### **Parameters**

HashMap		
String	of key	

## Return values

1=Exists,0=Does	not Exist

Here is the call graph for this function:



6.29.1.7 int hashMap\_ContainsValue ( struct hashMap \* hm, void \* val )

Check if hashmap contains a value.

HashMap	
---------	--

Value	to check for
Return values	
1-Evic	s 0-Dags   not Exist

Here is the call graph for this function:



6.29.1.8 struct hashMap\* hashMap\_Create ( unsigned int *initialEntries*, unsigned int *entryAllocationStep*, void \* clearItemFunction )

Create and allocate a hash map.

## **Parameters**

Number	of initial entry space
Allocation	step for new allocations
Pointer	to a function that clears an item

## Return values

Hashmap	Structure or , 0=Failure

Here is the call graph for this function:



6.29.1.9 void hashMap\_Destroy ( struct hashMap \* hm )

Destroy and deallocate a hash map.

HashMap	

Return values

```
1=Success,0=Failure
```

Here is the call graph for this function:



6.29.1.10 int hashMap\_FindIndex ( struct hashMap \* hm, char \* key, unsigned long \* index )

Find index of a key.

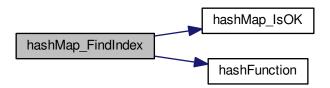
### **Parameters**

HashMap	
Input	String with the key index to find
Output	index of the record that holds the data we were searching for

## Return values

_		_
	1=Success,0=Failure	

Here is the call graph for this function:



 $\textbf{6.29.1.11} \quad \text{int hashMap\_GetCurrentNumberOfEntries ( struct hashMap} * \textit{hm} \ )$ 

Get the current number of entries of hash map.

**Parameters** 

HashMap	p
---------	---

Number of entries
-------------------

Here is the call graph for this function:



6.29.1.12 unsigned long hashMap\_GetHashAtIndex ( struct hashMap \* hm, unsigned int index )

Return key hash for index.

## **Parameters**

HashMap	
Index	number

## Return values

Hash	of key, or 0 for no key

Here is the call graph for this function:



6.29.1.13 char\* hashMap\_GetKeyAtIndex ( struct hashMap \* hm, unsigned int index )

Return key value for index.

## **Parameters**

HashMap	
Index	number

String of key, or 0 for	r no key
-------------------------	----------

Here is the call graph for this function:



6.29.1.14 int hashMap\_GetMaxNumberOfEntries ( struct hashMap \* hm )

Get the maximum number of entries of hash map.

#### **Parameters**

nasniviap
-----------

### Return values

Maximum	Number of entries

Here is the call graph for this function:



6.29.1.15 int hashMap\_GetPayload ( struct hashMap \* hm, char \* key, void \* payload )

Return payload for specified key.

### **Parameters**

HashMap	
Input	String of key
Output	Pointer of payload

1=Success,0=Failure
---------------------

Here is the call graph for this function:



6.29.1.16 int hashMap\_GetULongPayload ( struct hashMap \* hm, char \* key, unsigned long \* payload )

Return numerical payload for specified key.

#### **Parameters**

HashMap	
Input	String of key
Output	Pointer of payload

### Return values

1=Success,0=Failure	

Here is the call graph for this function:



6.29.1.17 int hashMap\_Grow ( struct hashMap \* hm, unsigned int growthSize )

Here is the call graph for this function:



6.29.1.18 int hashMap\_IsOK ( struct hashMap \* hm )

6.29.1.19 int hashMap\_lsSorted ( struct hashMap \*hm )

Here is the call graph for this function:



6.29.1.20 int hashMap\_LoadToFile ( struct hashMap \* hm, char \* filename )

Load hash map from a file.

#### **Parameters**

HashMap	structure
Filename	to save to

### **Return values**

1=Success,0=Fail	

Here is the call graph for this function:



6.29.1.21 int hashMap\_SaveToFile ( struct hashMap \* hm, char \* filename )

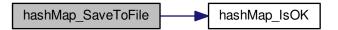
Save hash map to a file.

## **Parameters**

HashMap	structure
Filename	to save to

1=Success,0=Fail
------------------

Here is the call graph for this function:



6.29.1.22 int hashMap\_Sort ( struct hashMap \* hm )

Sort hash map.

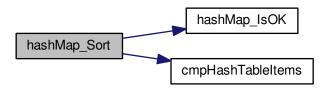
### **Parameters**

HashMap	

### **Return values**

```
1=Success,0=Failure
```

Here is the call graph for this function:



6.29.1.23 int hashmap\_SwapRecords ( struct hashMap \* hm, unsigned int index1, unsigned int index2 )

Swap two records.

### **Parameters**

HashMap	
Index	1 to be swapped
Index	2 to be swapped

1=Success,0=Failure
---------------------

Here is the call graph for this function:

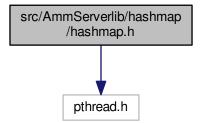


# 6.30 src/AmmServerlib/hashmap/hashmap.h File Reference

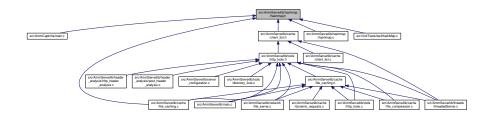
A uniform and clean way to create hashmaps in C and query them.

#include <pthread.h>

Include dependency graph for hashmap.h:



This graph shows which files directly or indirectly include this file:



## **Data Structures**

struct hashMapEntry

An entry on the hash map flattened out for ease of use.

struct hashMap

The central structure for the hash map.

### **Functions**

unsigned long hashFunction (char \*str)

The function that converts a string to a number so that it will be easier to be searched.

struct hashMap \* hashMap\_Create (unsigned int initialEntries, unsigned int entryAllocationStep, void \*clear-ltemFunction)

Create and allocate a hash map.

void hashMap\_Destroy (struct hashMap \*hm)

Destroy and deallocate a hash map.

int hashMap\_Sort (struct hashMap \*hm)

Sort hash map.

int hashMap\_Add (struct hashMap \*hm, char \*key, void \*val, unsigned int valLength)

Add a new key to hash map.

int hashMap\_AddULong (struct hashMap \*hm, char \*key, unsigned long val)

Add a new key (integer) to hash map.

int hashMap\_FindIndex (struct hashMap \*hm, char \*key, unsigned long \*index)

Find index of a key.

int hashmap\_SwapRecords (struct hashMap \*hm, unsigned int index1, unsigned int index2)

Swap two records.

char \* hashMap\_GetKeyAtIndex (struct hashMap \*hm, unsigned int index)

Return key value for index.

unsigned long hashMap\_GetHashAtIndex (struct hashMap \*hm, unsigned int index)

Return key hash for index.

int hashMap\_GetPayload (struct hashMap \*hm, char \*key, void \*payload)

Return payload for specified key.

• int hashMap\_GetULongPayload (struct hashMap \*hm, char \*key, unsigned long \*payload)

Return numerical payload for specified key.

void hashMap\_Clear (struct hashMap \*hm)

Clear all entries of hash map.

int hashMap\_ContainsKey (struct hashMap \*hm, char \*key)

Check if hashmap contains a key.

int hashMap\_ContainsValue (struct hashMap \*hm, void \*val)

Check if hashmap contains a value.

int hashMap\_GetMaxNumberOfEntries (struct hashMap \*hm)

Get the maximum number of entries of hash map.

int hashMap\_GetCurrentNumberOfEntries (struct hashMap \*hm)

Get the current number of entries of hash map.

int hashMap\_LoadToFile (struct hashMap \*hm, char \*filename)

Load hash map from a file.

int hashMap\_SaveToFile (struct hashMap \*hm, char \*filename)

Save hash map to a file.

## 6.30.1 Detailed Description

A uniform and clean way to create hashmaps in C and query them.

Author

Ammar Qammaz (AmmarkoV)

Bug This hashmap implementation uses serial searches for now, and needs a lot of work

## 6.30.2 Function Documentation

## 6.30.2.1 unsigned long hashFunction ( char \* str )

The function that converts a string to a number so that it will be easier to be searched.

djb2 This algorithm (k=33) was first reported by dan bernstein many years ago in comp.lang.c. another version of this algorithm (now favored by bernstein) uses xor: hash(i) = hash(i - 1) \* 33  $^{\land}$  str[i]; the magic of number 33 (why it works better than many other constants, prime or not) has never been adequately explained. Needless to say , this is our hash function..!

6.30.2.2 int hashMap\_Add ( struct hashMap \* hm, char \* key, void \* val, unsigned int valLength )

Add a new key to hash map.

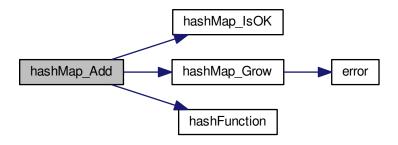
#### **Parameters**

HashMap	
String	with the key index
String	with the value of this record
Length	of the value

#### Return values

1=Success,0=Failure	
---------------------	--

Here is the call graph for this function:



6.30.2.3 int hashMap\_AddULong ( struct hashMap \* hm, char \* key, unsigned long val )

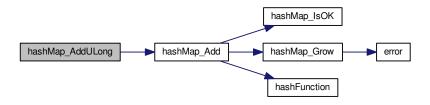
Add a new key (integer) to hash map.

HashMap	
String	with the key index
Number	value of this record

**Return values** 

1=Success,0=Failure

Here is the call graph for this function:



6.30.2.4 void hashMap\_Clear ( struct hashMap \* hm )

Clear all entries of hash map.

**Parameters** 

HashMap	

Return values

No	return value

Here is the call graph for this function:



6.30.2.5 int hashMap\_ContainsKey ( struct hashMap \* hm, char \* key )

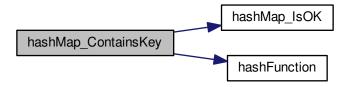
Check if hashmap contains a key.

**Parameters** 

HashMap	
String	of key

1=Exists,0=Does	not Exist

Here is the call graph for this function:



6.30.2.6 int hashMap\_ContainsValue ( struct hashMap \* hm, void \* val )

Check if hashmap contains a value.

### **Parameters**

HashMap	
Value	to check for

## Return values

1=Exists,0=Does	not Exist

Here is the call graph for this function:



6.30.2.7 struct hashMap\* hashMap\_Create ( unsigned int *initialEntries*, unsigned int *entryAllocationStep*, void \* clearItemFunction )

Create and allocate a hash map.

Number	of initial entry space
Allocation	step for new allocations
Pointer	to a function that clears an item

Return values

Hashmap	Structure or , 0=Failure

Here is the call graph for this function:



6.30.2.8 void hashMap\_Destroy ( struct hashMap \* hm )

Destroy and deallocate a hash map.

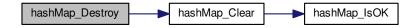
**Parameters** 

HashMap |

**Return values** 

1=Success,0=Failure

Here is the call graph for this function:



6.30.2.9 int hashMap\_FindIndex ( struct hashMap \* hm, char \* key, unsigned long \* index )

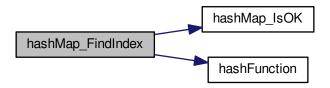
Find index of a key.

### **Parameters**

HashMap	
Input	String with the key index to find
Output	index of the record that holds the data we were searching for

1=Success,0=Failure
---------------------

Here is the call graph for this function:



 $6.30.2.10 \quad int\ hashMap\_GetCurrentNumberOfEntries (\ struct\ hashMap* \textit{hm}\ )$ 

Get the current number of entries of hash map.

#### **Parameters**

### Return values

Number of entries
-------------------

Here is the call graph for this function:



6.30.2.11 unsigned long hashMap\_GetHashAtIndex ( struct hashMap \* hm, unsigned int index )

Return key hash for index.

# **Parameters**

HashMap	
Index	number

Hash of key, or 0 for no key	
------------------------------	--

Here is the call graph for this function:



6.30.2.12 char\* hashMap\_GetKeyAtIndex ( struct hashMap \* hm, unsigned int index )

Return key value for index.

### **Parameters**

HashMap	
Index	number

### Return values

String	of key, or 0 for no key

Here is the call graph for this function:



6.30.2.13 int hashMap\_GetMaxNumberOfEntries ( struct hashMap \* hm )

Get the maximum number of entries of hash map.

# **Parameters**

HashMap	p
---------	---

Maximum	Number of entries
---------	-------------------

Here is the call graph for this function:



6.30.2.14 int hashMap\_GetPayload ( struct hashMap \* hm, char \* key, void \* payload )

Return payload for specified key.

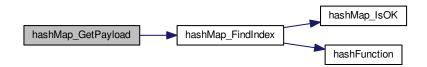
### **Parameters**

HashMap	
Input	String of key
Output	Pointer of payload

### Return values

1 Cussess O Failure	
i=Success,∪=raiiure	

Here is the call graph for this function:



 $6.30.2.15 \quad \text{int hashMap\_GetULongPayload ( struct hashMap} * \textit{hm}, \ \text{char} * \textit{key}, \ \text{unsigned long} * \textit{payload} \ )$ 

Return numerical payload for specified key.

### **Parameters**

HashMap	
Input	String of key
Output	Pointer of payload

1=Success,0=Failure
---------------------

Here is the call graph for this function:



6.30.2.16 int hashMap\_LoadToFile ( struct hashMap \* hm, char \* filename )

Load hash map from a file.

### **Parameters**

HashMap	structure
Filename	to save to

### Return values

1=Success,0=Fail	

Here is the call graph for this function:



6.30.2.17 int hashMap\_SaveToFile ( struct hashMap \* hm, char \* filename )

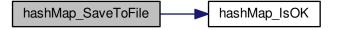
Save hash map to a file.

# **Parameters**

HashMap	structure
Filename	to save to

1_Success 0_Eail	
1=30000035,0=1 all	
	1

Here is the call graph for this function:



6.30.2.18 int hashMap\_Sort ( struct hashMap \* hm )

Sort hash map.

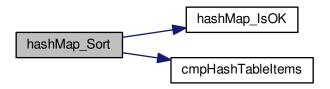
### **Parameters**

HashMap	

### Return values

```
1=Success,0=Failure
```

Here is the call graph for this function:



 $6.30.2.19 \quad \text{int hashmap\_SwapRecords ( struct hashMap}* \textit{hm, unsigned int index1, unsigned int index2} \ )$ 

Swap two records.

### **Parameters**

HashMap	
Index	1 to be swapped
Index	2 to be swapped

1=Success,0=Failure
---------------------

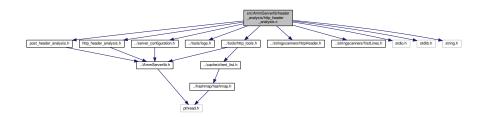
Here is the call graph for this function:



# 6.31 src/AmmServerlib/header\_analysis/http\_header\_analysis.c File Reference

```
#include "http_header_analysis.h"
#include "post_header_analysis.h"
#include "../tools/http_tools.h"
#include "../tools/logs.h"
#include "../server_configuration.h"
#include "../stringscanners/httpHeader.h"
#include "../stringscanners/firstLines.h"
#include <stdio.h>
#include <stdib.h>
#include <string.h>
```

Include dependency graph for http\_header\_analysis.c:



# Macros

- #define CR 13
- #define LF 10

# **Functions**

 char \* ReceiveHTTPHeader (struct AmmServer\_Instance \*instance, int clientSock, unsigned long \*header-Length)

Receive an HTTP Header from a socket and prepare it for further processing.

• int AppendPOSTRequestToHTTPHeader (struct HTTPTransaction \*transaction)

POST requests also have a payload appended that we consider part of the whole "header" so we need to keep on reading it..!

• int FreeHTTPHeader (struct HTTPHeader \*output)

Deallocate memory occupied by an HTTP Header.

• int HTTPHeaderComplete (char \*request, unsigned int request\_length)

Ask if a header is complete inside an incoming request, detected by four consecutive bytes CR LF CR LF that mark the end of a header.

- int HTTPHeaderIsPOST (char \*request, unsigned int requestLength)
  - Ask if a header is a POST request, detected by the first four consecutive bytes being P O S T.
- int ProcessFirstHTTPLine (struct HTTPHeader \*output, char \*request, unsigned int request\_length, char \*webserver\_root)
- int ProcessAuthorizationHTTPLine (struct AmmServer\_Instance \*instance, struct HTTPHeader \*output, char \*request, unsigned int request\_length, unsigned int \*payload\_pos)
- int ProcessRangeHTTPLine (char \*request, unsigned int requestLength, unsigned long \*rangeStart, unsigned long \*rangeEnd)
- int AnalyzeHTTPLineRequest (struct AmmServer\_Instance \*instance, struct HTTPHeader \*output, char \*request, unsigned int request\_length, unsigned int lines\_gathered, char \*webserver\_root)
- int AnalyzeHTTPHeader (struct AmmServer\_Instance \*instance, struct HTTPTransaction \*transaction)

Analyze HTTP header ( after it has been accumulated into memory )

#### 6.31.1 Macro Definition Documentation

- 6.31.1.1 #define CR 13
- 6.31.1.2 #define LF 10

### 6.31.2 Function Documentation

6.31.2.1 int AnalyzeHTTPHeader ( struct AmmServer Instance \* instance, struct HTTPTransaction \* transaction )

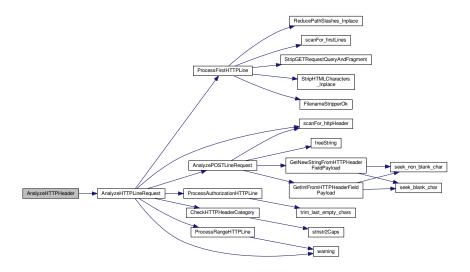
Analyze HTTP header ( after it has been accumulated into memory )

#### **Parameters**

An	AmmarServer Instance
HTTP-	we are talking about
Transaction	

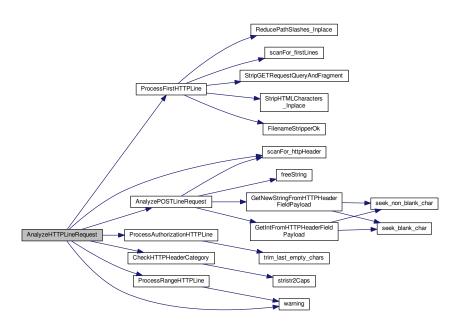
### Return values

1=Success,0=Failure	



6.31.2.2 int AnalyzeHTTPLineRequest ( struct AmmServer\_Instance \* instance, struct HTTPHeader \* output, char \* request, unsigned int request\_length, unsigned int lines\_gathered, char \* webserver\_root )

Here is the call graph for this function:



### 6.31.2.3 int AppendPOSTRequestToHTTPHeader ( struct HTTPTransaction \* transaction )

POST requests also have a payload appended that we consider part of the whole "header" so we need to keep on reading it..!

#### **Parameters**

```
HTTP-
Transaction
```

# Return values

```
1=Success,0=Failure
```

Here is the call graph for this function:



### 6.31.2.4 int FreeHTTPHeader ( struct HTTPHeader \* output )

Deallocate memory occupied by an HTTP Header.

#### **Parameters**

		_
HTTPHeader	to be deallocated	7

#### Return values

```
1=Success,0=Failure
```

6.31.2.5 int HTTPHeaderComplete ( char \* request, unsigned int request\_length )

Ask if a header is complete inside an incoming request , detected by four consecutive bytes CR LF CR LF that mark the end of a header.

#### **Parameters**

Pointer	to incoming request (streaming) string
Length	of incoming string

#### Return values

1=Complete,0=Incomplete	

6.31.2.6 int HTTPHeaderIsPOST ( char \* request, unsigned int requestLength )

Ask if a header is a POST request, detected by the first four consecutive bytes being POST.

#### **Parameters**

Pointer	to incoming request (streaming) string
Length	of incoming string

### Return values

|--|

6.31.2.7 int ProcessAuthorizationHTTPLine ( struct AmmServer\_Instance \* instance, struct HTTPHeader \* output, char \* request, unsigned int request\_length, unsigned int \* payload\_pos ) [inline]

Here is the call graph for this function:

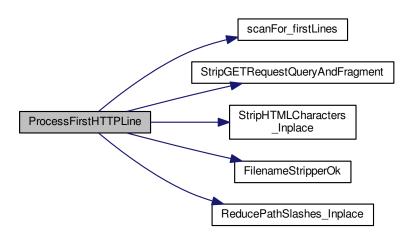


6.31.2.8 int ProcessFirstHTTPLine ( struct HTTPHeader \* output, char \* request, unsigned int request\_length, char \* webserver\_root )

Input String Verification from client Since this string will be passed to fopen this can be dangerous so we perform some security checks with FilenameStripperOk to make sure no escape characters subdirs out of public\_html (via public\_html/./etc) and overflows may happen..! Most of the functions are implemented in http\_tools! The results

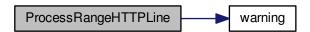
are then copied to output->resource and output->verified\_local\_resource which contain the resource requested as the client stated it and as we verified for local filesystem..!

Here is the call graph for this function:



6.31.2.9 int ProcessRangeHTTPLine ( char \* request, unsigned int requestLength, unsigned long \* rangeStart, unsigned long \* rangeEnd ) [inline]

Here is the call graph for this function:



6.31.2.10 char\* ReceiveHTTPHeader ( struct AmmServer\_Instance \* instance, int clientSock, unsigned long \* headerLength )

Receive an HTTP Header from a socket and prepare it for further processing.

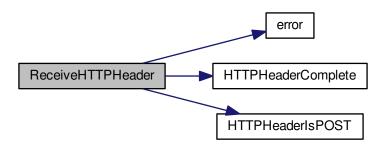
### **Parameters**

An	AmmarServer Instance
Socket	that we should recv to get the http header
Output	length of incoming header

Return values

Pointer	to memory containing HTTPHeader,0=Failure

Here is the call graph for this function:

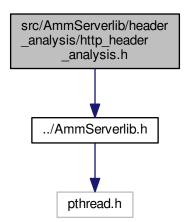


# 6.32 src/AmmServerlib/header\_analysis/http\_header\_analysis.h File Reference

Tools to process HTTP requests.

#include "../AmmServerlib.h"

Include dependency graph for http\_header\_analysis.h:



This graph shows which files directly or indirectly include this file:



#### **Functions**

 char \* ReceiveHTTPHeader (struct AmmServer\_Instance \*instance, int clientSock, unsigned long \*header-Length)

Receive an HTTP Header from a socket and prepare it for further processing.

• int AppendPOSTRequestToHTTPHeader (struct HTTPTransaction \*transaction)

POST requests also have a payload appended that we consider part of the whole "header" so we need to keep on reading it..!

• int FreeHTTPHeader (struct HTTPHeader \*output)

Deallocate memory occupied by an HTTP Header.

int HTTPHeaderComplete (char \*request, unsigned int request length)

Ask if a header is complete inside an incoming request, detected by four consecutive bytes CR LF CR LF that mark the end of a header.

• int HTTPHeaderIsPOST (char \*request, unsigned int requestLength)

Ask if a header is a POST request, detected by the first four consecutive bytes being P O S T.

• int AnalyzeHTTPHeader (struct AmmServer\_Instance \*instance, struct HTTPTransaction \*transaction)

Analyze HTTP header ( after it has been accumulated into memory )

### 6.32.1 Detailed Description

Tools to process HTTP requests.

**Author** 

Ammar Qammaz (AmmarkoV)

**Bug** HTTP header analysis can be improved (code style etc.) although the recent use of stringscanners has greatly improved it and reduced lines of code

### 6.32.2 Function Documentation

6.32.2.1 int AnalyzeHTTPHeader ( struct AmmServer Instance \* instance, struct HTTPTransaction \* transaction )

Analyze HTTP header ( after it has been accumulated into memory )

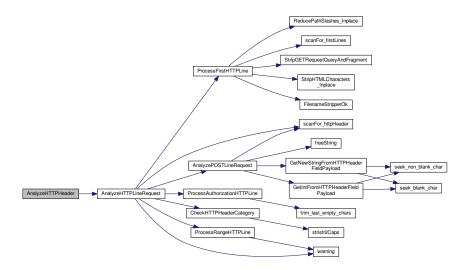
### Parameters

An	AmmarServer Instance
HTTP-	we are talking about
Transaction	

Return values

```
1=Success,0=Failure
```

Here is the call graph for this function:



### 6.32.2.2 int AppendPOSTRequestToHTTPHeader ( struct HTTPTransaction \* transaction )

POST requests also have a payload appended that we consider part of the whole "header" so we need to keep on reading it..!

**Parameters** 

```
HTTP-
Transaction
```

Return values

```
1=Success,0=Failure
```

Here is the call graph for this function:



# 6.32.2.3 int FreeHTTPHeader ( struct HTTPHeader \* output )

Deallocate memory occupied by an HTTP Header.

#### **Parameters**

HTTPHeader	to be deallocated
------------	-------------------

### **Return values**

1=Success,0=Failure	

### 6.32.2.4 int HTTPHeaderComplete ( char \* request, unsigned int request\_length )

Ask if a header is complete inside an incoming request , detected by four consecutive bytes CR LF CR LF that mark the end of a header.

#### **Parameters**

Pointer	to incoming request (streaming) string
Length	of incoming string

### Return values

1=Complete.0=Incomplete	

### 6.32.2.5 int HTTPHeaderIsPOST ( char \* request, unsigned int requestLength )

Ask if a header is a POST request, detected by the first four consecutive bytes being POST.

# Parameters

Pointer	to incoming request (streaming) string
Length	of incoming string

### Return values

1=POST,0=Not   POST request
-----------------------------

6.32.2.6 char\* ReceiveHTTPHeader ( struct AmmServer\_Instance \* instance, int clientSock, unsigned long \* headerLength )

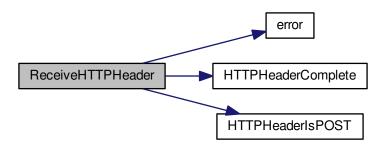
Receive an HTTP Header from a socket and prepare it for further processing.

### **Parameters**

An	AmmarServer Instance
Socket	that we should recv to get the http header
Output	length of incoming header

Pointer	to memory containing HTTPHeader,0=Failure
---------	---

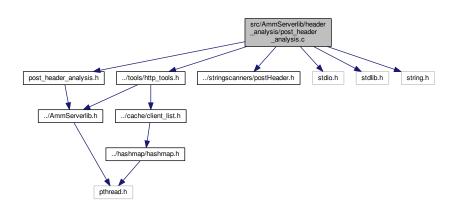
Here is the call graph for this function:



# 6.33 src/AmmServerlib/header\_analysis/post\_header\_analysis.c File Reference

```
#include "post_header_analysis.h"
#include "../tools/http_tools.h"
#include "../stringscanners/postHeader.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

Include dependency graph for post\_header\_analysis.c:



### **Functions**

• int AnalyzePOSTLineRequest (struct AmmServer\_Instance \*instance, struct HTTPHeader \*output, char \*request, unsigned int request\_length, unsigned int lines\_gathered, char \*webserver\_root)

Analyze a POST request line by line filling in the structures that define it.

# 6.33.1 Function Documentation

6.33.1.1 int AnalyzePOSTLineRequest ( struct AmmServer\_Instance \* instance, struct HTTPHeader \* output, char \* request, unsigned int request\_length, unsigned int lines\_gathered, char \* webserver\_root )

Analyze a POST request line by line filling in the structures that define it.

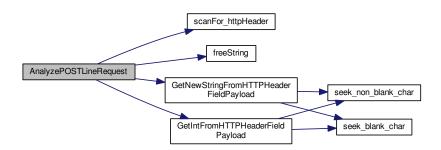
#### **Parameters**

An	AmmarServer Instance
Output	HTTPHeader with information
Memory	block with the incoming request
Length	of Memory block of the incoming request
Current	line we are at
Filename	of web server root directory ( public_html )

### Return values

1=Success,0=Failure	

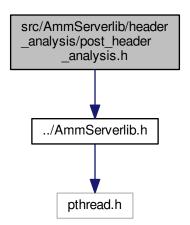
Here is the call graph for this function:



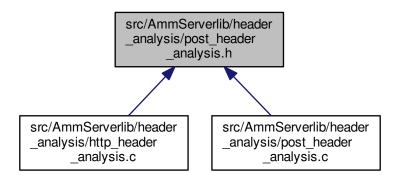
# 6.34 src/AmmServerlib/header\_analysis/post\_header\_analysis.h File Reference

Tools to process POST requests.

#include "../AmmServerlib.h"
Include dependency graph for post\_header\_analysis.h:



This graph shows which files directly or indirectly include this file:



### **Functions**

• int AnalyzePOSTLineRequest (struct AmmServer\_Instance \*instance, struct HTTPHeader \*output, char \*request, unsigned int request\_length, unsigned int lines\_gathered, char \*webserver\_root)

Analyze a POST request line by line filling in the structures that define it.

### 6.34.1 Detailed Description

Tools to process POST requests.

**Author** 

Ammar Qammaz (AmmarkoV)

Bug POST header analysis is not fully implemented yet

#### 6.34.2 Function Documentation

6.34.2.1 int AnalyzePOSTLineRequest ( struct AmmServer\_Instance \* instance, struct HTTPHeader \* output, char \* request, unsigned int request\_length, unsigned int lines\_gathered, char \* webserver\_root )

Analyze a POST request line by line filling in the structures that define it.

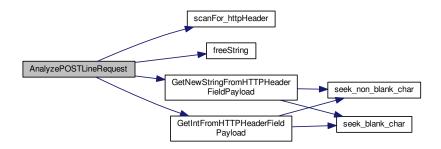
### **Parameters**

An	AmmarServer Instance
Output	HTTPHeader with information
Memory	block with the incoming request
Length	of Memory block of the incoming request
Current	line we are at
Filename	of web server root directory ( public_html )

Return values

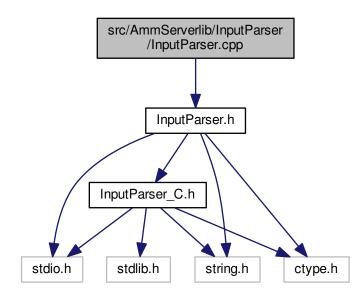
```
1=Success,0=Failure
```

Here is the call graph for this function:



# 6.35 src/AmmServerlib/InputParser/InputParser.cpp File Reference

#include "InputParser.h"
Include dependency graph for InputParser.cpp:



# **Functions**

• const char \* Version ()

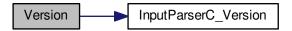
# **Variables**

```
    const char * ver =" VERSION 1.30 - 7/1/10 \0"
```

### 6.35.1 Function Documentation

```
6.35.1.1 const char* Version ( )
```

Here is the call graph for this function:



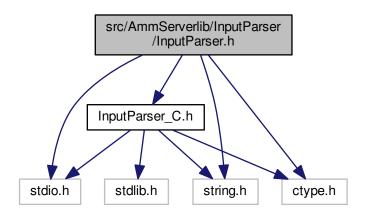
# 6.35.2 Variable Documentation

6.35.2.1 const char\* ver =" VERSION 1.30 - 7/1/10 \0"

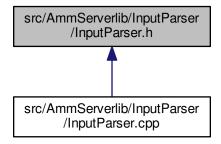
# 6.36 src/AmmServerlib/InputParser/InputParser.h File Reference

```
#include "InputParser_C.h"
#include <stdio.h>
#include <string.h>
#include <ctype.h>
```

Include dependency graph for InputParser.h:



This graph shows which files directly or indirectly include this file:

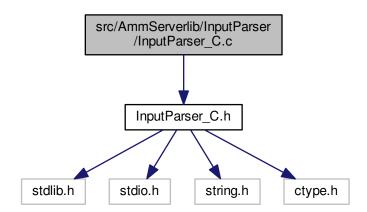


# **Data Structures**

· class InputParser

# 6.37 src/AmmServerlib/InputParser/InputParser\_C.c File Reference

#include "InputParser\_C.h"
Include dependency graph for InputParser\_C.c:



### **Macros**

• #define WARN\_ABOUT\_INCORRECTLY\_ALLOCATED\_STACK\_STRINGS 1

#### **Functions**

- char \* InputParserC Version ()
- int InputParser\_ClearNonCharacters (char \*inpt, unsigned int length)
- int InputParser TrimCharactersStart (char \*inpt, unsigned int length, char what2trim)
- int InputParser\_TrimCharactersEnd (char \*inpt, unsigned int length, char what2trim)
- int InputParser\_TrimCharacters (char \*inpt, unsigned int length, char what2trim)
- signed int Str2Int internal (char \*inpt, unsigned int start from, unsigned int length)
- unsigned char CheckIPCOk (struct InputParserC \*ipc)
- void InputParser DefaultDelimeters (struct InputParserC \*ipc)
- struct InputParserC \* InputParser\_Create (unsigned int max\_string\_count, unsigned int max\_delimiter\_count)
- void InputParser\_Destroy (struct InputParserC \*ipc)
- unsigned char CheckDelimeterNumOk (struct InputParserC \*ipc, int num)
- void InputParser SetDelimeter (struct InputParserC \*ipc, int num, char tmp)
- char InputParser\_GetDelimeter (struct InputParserC \*ipc, int num)
- unsigned char InputParser SelfCheck (struct InputParserC \*ipc)
- unsigned char CheckWordNumOk (struct InputParserC \*ipc, unsigned int num)
- unsigned int InputParser\_GetWord (struct InputParserC \*ipc, unsigned int num, char \*wheretostore, unsigned int storagesize)
- unsigned char InputParser\_WordCompareNoCase (struct InputParserC \*ipc, unsigned int num, char \*word, unsigned int wordsize)
- unsigned char InputParser\_WordCompareNoCaseAuto (struct InputParserC \*ipc, unsigned int num, char \*word)
- unsigned char InputParser\_WordCompare (struct InputParserC \*ipc, unsigned int num, char \*word, unsigned int wordsize)
- unsigned char InputParser\_WordCompareAuto (struct InputParserC \*ipc, unsigned int num, char \*word)
- unsigned int InputParser\_GetUpcaseWord (struct InputParserC \*ipc, unsigned int num, char \*wheretostore, unsigned int storagesize)
- unsigned int InputParser\_GetLowercaseWord (struct InputParserC \*ipc, unsigned int num, char \*wheretostore, unsigned int storagesize)
- char InputParser GetWordChar (struct InputParserC \*ipc, unsigned int num, unsigned int pos)
- signed int InputParser\_GetWordInt (struct InputParserC \*ipc, unsigned int num)
- float InputParser\_GetWordFloat (struct InputParserC \*ipc, unsigned int num)
- unsigned int InputParser\_GetWordLength (struct InputParserC \*ipc, unsigned int num)
- int InputParser\_SeperateWords (struct InputParserC \*ipc, char \*inpt, char keepcopy)
- int InputParser\_SeperateWordsCC (struct InputParserC \*ipc, const char \*inpt, char keepcopy)
- int InputParser\_SeperateWordsUC (struct InputParserC \*ipc, unsigned char \*inpt, char keepcopy)

### **Variables**

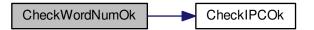
- int warningsAboutIncorrectlyAllocatedStackIssued = 0
- char \_ipc\_ver [] =" 0.356 written from scratch 8/2/10 \0"

### 6.37.1 Macro Definition Documentation

- 6.37.1.1 #define WARN\_ABOUT\_INCORRECTLY\_ALLOCATED\_STACK\_STRINGS 1
- 6.37.2 Function Documentation
- 6.37.2.1 unsigned char CheckDelimeterNumOk ( struct InputParserC \* ipc, int num ) [inline]
- **6.37.2.2** unsigned char ChecklPCOk ( struct InputParserC \* *ipc* ) [inline]

6.37.2.3 unsigned char CheckWordNumOk ( struct InputParserC \* ipc, unsigned int num ) [inline]

Here is the call graph for this function:



6.37.2.4 int InputParser\_ClearNonCharacters ( char \* inpt, unsigned int length )

6.37.2.5 struct InputParserC\* InputParser\_Create ( unsigned int max\_string\_count, unsigned int max\_delimiter\_count )

Here is the call graph for this function:



6.37.2.6 void InputParser\_DefaultDelimeters ( struct InputParserC \* ipc )

Here is the call graph for this function:



6.37.2.7 void InputParser\_Destroy ( struct InputParserC \* ipc )

6.37.2.8 char InputParser\_GetDelimeter ( struct InputParserC \* ipc, int num )

Here is the call graph for this function:



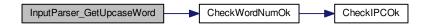
6.37.2.9 unsigned int InputParser\_GetLowercaseWord ( struct InputParserC \* ipc, unsigned int num, char \* wheretostore, unsigned int storagesize )

Here is the call graph for this function:



6.37.2.10 unsigned int InputParser\_GetUpcaseWord ( struct InputParserC \* ipc, unsigned int num, char \* wheretostore, unsigned int storagesize )

Here is the call graph for this function:

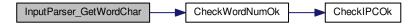


6.37.2.11 unsigned int InputParser\_GetWord ( struct InputParserC \* ipc, unsigned int num, char \* wheretostore, unsigned int storagesize )



6.37.2.12 char InputParser\_GetWordChar ( struct InputParserC \* ipc, unsigned int num, unsigned int pos )

Here is the call graph for this function:



6.37.2.13 float InputParser\_GetWordFloat ( struct InputParserC \* ipc, unsigned int num )

Here is the call graph for this function:



6.37.2.14 signed int InputParser\_GetWordInt ( struct InputParserC \* ipc, unsigned int num )

Here is the call graph for this function:



6.37.2.15 unsigned int InputParser\_GetWordLength ( struct InputParserC \* ipc, unsigned int num )



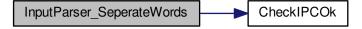
6.37.2.16 unsigned char InputParser\_SelfCheck ( struct InputParserC \* ipc )

Here is the call graph for this function:



6.37.2.17 int InputParser\_SeperateWords ( struct InputParserC \* ipc, char \* inpt, char keepcopy )

Here is the call graph for this function:



6.37.2.18 int InputParser\_SeperateWordsCC ( struct InputParserC \* ipc, const char \* inpt, char keepcopy )

Here is the call graph for this function:



6.37.2.19 int InputParser\_SeperateWordsUC ( struct InputParserC \* ipc, unsigned char \* inpt, char keepcopy )



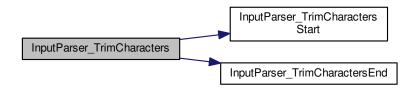
6.37.2.20 void InputParser\_SetDelimeter ( struct InputParserC \* ipc, int num, char tmp )

Here is the call graph for this function:



6.37.2.21 int InputParser\_TrimCharacters ( char \* inpt, unsigned int length, char what2trim )

Here is the call graph for this function:



- 6.37.2.22 int InputParser\_TrimCharactersEnd ( char \* inpt, unsigned int length, char what2trim )
- 6.37.2.23 int InputParser\_TrimCharactersStart ( char \* inpt, unsigned int length, char what2trim )
- 6.37.2.24 unsigned char InputParser\_WordCompare ( struct InputParserC \* ipc, unsigned int num, char \* word, unsigned int wordsize )



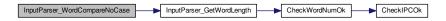
6.37.2.25 unsigned char InputParser\_WordCompareAuto ( struct InputParserC \* ipc, unsigned int num, char \* word )

Here is the call graph for this function:



6.37.2.26 unsigned char InputParser\_WordCompareNoCase ( struct InputParserC \* ipc, unsigned int num, char \* word, unsigned int wordsize )

Here is the call graph for this function:



6.37.2.27 unsigned char InputParser\_WordCompareNoCaseAuto ( struct InputParserC \* ipc, unsigned int num, char \* word )

Here is the call graph for this function:

```
InputParser_WordCompareNo CaseAuto InputParser_WordCompareNoCase InputParser_GetWordLength CheckWordNumOk CheckIPCOk
```

```
6.37.2.28 char* InputParserC_Version ( )
```

6.37.2.29 signed int Str2Int\_internal ( char \* inpt, unsigned int start\_from, unsigned int length ) [inline]

### 6.37.3 Variable Documentation

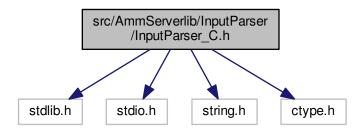
6.37.3.1 char \_ipc\_ver[] =" 0.356 written from scratch - 8/2/10 \0"

6.37.3.2 int warningsAboutIncorrectlyAllocatedStackIssued = 0

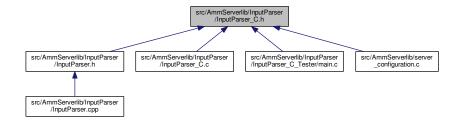
# 6.38 src/AmmServerlib/InputParser/InputParser\_C.h File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <ctype.h>
```

Include dependency graph for InputParser\_C.h:



This graph shows which files directly or indirectly include this file:



### **Data Structures**

- · struct tokens
- · struct guard\_byte
- struct InputParserC

### **Macros**

- #define DELIM\_MAX\_MAX 6
- #define CONTAINERS\_MAX 1
- #define MAX\_COMPLICITY 4
- #define MAX\_MEMORY 256
- #define MAX\_STRING 2048

### **Functions**

- char \* InputParserC\_Version ()
- int InputParser\_ClearNonCharacters (char \*inpt, unsigned int length)
- int InputParser\_TrimCharactersStart (char \*inpt, unsigned int length, char what2trim)
- int InputParser\_TrimCharactersEnd (char \*inpt, unsigned int length, char what2trim)
- int InputParser\_TrimCharacters (char \*inpt, unsigned int length, char what2trim)
- void InputParser\_DefaultDelimeters (struct InputParserC \*ipc)

- void InputParser\_SetDelimeter (struct InputParserC \*ipc, int num, char tmp)
- char InputParser\_GetDelimeter (struct InputParserC \*ipc, int num)
- struct InputParserC \* InputParser Create (unsigned int max string count, unsigned int max delimiter count)
- void InputParser Destroy (struct InputParserC \*ipc)
- unsigned char InputParser SelfCheck (struct InputParserC \*ipc)
- unsigned char CheckWordNumOk (struct InputParserC \*ipc, unsigned int num)
- char InputParser\_GetWordChar (struct InputParserC \*ipc, unsigned int num, unsigned int pos)
- unsigned char InputParser\_WordCompareNoCase (struct InputParserC \*ipc, unsigned int num, char \*word, unsigned int wordsize)
- unsigned char InputParser\_WordCompareNoCaseAuto (struct InputParserC \*ipc, unsigned int num, char \*word)
- unsigned char InputParser\_WordCompare (struct InputParserC \*ipc, unsigned int num, char \*word, unsigned int wordsize)
- unsigned char InputParser WordCompareAuto (struct InputParserC \*ipc, unsigned int num, char \*word)
- unsigned int InputParser\_GetWord (struct InputParserC \*ipc, unsigned int num, char \*wheretostore, unsigned int storagesize)
- unsigned int InputParser\_GetUpcaseWord (struct InputParserC \*ipc, unsigned int num, char \*wheretostore, unsigned int storagesize)
- unsigned int InputParser\_GetLowercaseWord (struct InputParserC \*ipc, unsigned int num, char \*wheretostore, unsigned int storagesize)
- signed int InputParser\_GetWordInt (struct InputParserC \*ipc, unsigned int num)
- float InputParser\_GetWordFloat (struct InputParserC \*ipc, unsigned int num)
- unsigned int InputParser GetWordLength (struct InputParserC \*ipc, unsigned int num)
- int InputParser SeperateWords (struct InputParserC \*ipc, char \*inpt, char keepcopy)
- int InputParser\_SeperateWordsCC (struct InputParserC \*ipc, const char \*inpt, char keepcopy)
- int InputParser\_SeperateWordsUC (struct InputParserC \*ipc, unsigned char \*inpt, char keepcopy)

### 6.38.1 Macro Definition Documentation

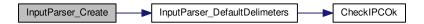
- 6.38.1.1 #define CONTAINERS\_MAX 1
- 6.38.1.2 #define DELIM\_MAX\_MAX 6
- 6.38.1.3 #define MAX\_COMPLICITY 4
- 6.38.1.4 #define MAX\_MEMORY 256
- 6.38.1.5 #define MAX\_STRING 2048
- 6.38.2 Function Documentation
- $\textbf{6.38.2.1} \quad unsigned \ char \ \textbf{CheckWordNumOk} \ ( \ \textbf{struct InputParserC} * \textit{ipc, unsigned int num } ) \quad [\texttt{inline}]$



6.38.2.2 int InputParser\_ClearNonCharacters ( char \* inpt, unsigned int length )

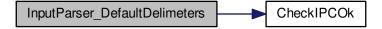
6.38.2.3 struct InputParserC\* InputParser\_Create ( unsigned int max\_string\_count, unsigned int max\_delimiter\_count )

Here is the call graph for this function:



6.38.2.4 void InputParser\_DefaultDelimeters ( struct InputParserC \* ipc )

Here is the call graph for this function:



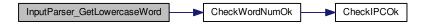
 $\textbf{6.38.2.5} \quad \text{void InputParser\_Destroy ( struct InputParserC} * \textit{ipc} \ )$ 

6.38.2.6 char InputParser\_GetDelimeter ( struct InputParserC \* ipc, int num )



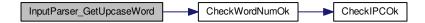
6.38.2.7 unsigned int InputParser\_GetLowercaseWord ( struct InputParserC \* ipc, unsigned int num, char \* wheretostore, unsigned int storagesize )

Here is the call graph for this function:



6.38.2.8 unsigned int InputParser\_GetUpcaseWord ( struct InputParserC \* ipc, unsigned int num, char \* wheretostore, unsigned int storagesize )

Here is the call graph for this function:

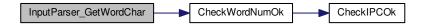


6.38.2.9 unsigned int InputParser\_GetWord ( struct InputParserC \* ipc, unsigned int num, char \* wheretostore, unsigned int storagesize )

Here is the call graph for this function:



6.38.2.10 char InputParser\_GetWordChar ( struct InputParserC \* ipc, unsigned int num, unsigned int pos )



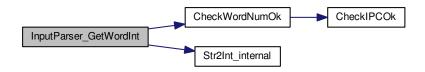
6.38.2.11 float InputParser\_GetWordFloat ( struct InputParserC \* ipc, unsigned int num )

Here is the call graph for this function:



6.38.2.12 signed int InputParser\_GetWordInt ( struct InputParserC \* ipc, unsigned int num )

Here is the call graph for this function:



6.38.2.13 unsigned int InputParser\_GetWordLength ( struct InputParserC \* ipc, unsigned int num )

Here is the call graph for this function:



6.38.2.14 unsigned char InputParser\_SelfCheck ( struct InputParserC \* ipc )



6.38.2.15 int InputParser\_SeperateWords ( struct InputParserC \* ipc, char \* inpt, char keepcopy )

Here is the call graph for this function:



6.38.2.16 int InputParser\_SeperateWordsCC ( struct InputParserC \* ipc, const char \* inpt, char keepcopy )

Here is the call graph for this function:



6.38.2.17 int InputParser\_SeperateWordsUC ( struct InputParserC \* ipc, unsigned char \* inpt, char keepcopy )

Here is the call graph for this function:

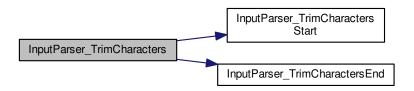


6.38.2.18 void InputParser\_SetDelimeter ( struct InputParserC \* ipc, int num, char tmp )



6.38.2.19 int InputParser\_TrimCharacters ( char \* inpt, unsigned int length, char what2trim )

Here is the call graph for this function:



- 6.38.2.20 int InputParser\_TrimCharactersEnd ( char \* inpt, unsigned int length, char what2trim )
- 6.38.2.21 int InputParser\_TrimCharactersStart ( char \* inpt, unsigned int length, char what2trim )
- 6.38.2.22 unsigned char InputParser\_WordCompare ( struct InputParserC \* ipc, unsigned int num, char \* word, unsigned int wordsize )

Here is the call graph for this function:



6.38.2.23 unsigned char InputParser\_WordCompareAuto ( struct InputParserC \* ipc, unsigned int num, char \* word )

Here is the call graph for this function:



6.38.2.24 unsigned char InputParser\_WordCompareNoCase ( struct InputParserC \* ipc, unsigned int num, char \* word, unsigned int wordsize )



6.38.2.25 unsigned char InputParser\_WordCompareNoCaseAuto ( struct InputParserC \* ipc, unsigned int num, char \* word )

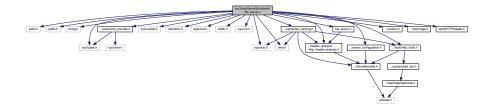
Here is the call graph for this function:

```
InputParser_WordCompareNo CaseAutio InputParser_WordCompareNoCase InputParser_GetWordLength CheckWordNumCk CheckIPCOk
```

```
6.38.2.26 char* InputParserC_Version ( )
```

## 6.39 src/AmmServerlib/network/file server.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <sys/uio.h>
#include <sys/stat.h>
#include <time.h>
#include "../version.h"
#include "file_server.h"
#include "../cache/file_caching.h"
#include "../header_analysis/http_header_analysis.h"
#include "../server_configuration.h"
#include "../tools/http_tools.h"
#include "../tools/time_provider.h"
#include "../tools/logs.h"
#include "sendHTTPHeader.h"
Include dependency graph for file_server.c:
```



## **Functions**

- int SendPart (int clientsock, char \*message, unsigned int message size)
- int TransmitFileToSocketInternal (FILE \*pFile, int clientsock, unsigned long bytesToSendStart)
- int TransmitFileToSocket (int clientsock, char \*verified\_filename, unsigned long start\_at\_byte, unsigned long end\_at\_byte)
- unsigned long SendFile (struct AmmServer\_Instance \*instance, struct HTTPTransaction \*transaction, char \*verified\_filename\_pending\_copy, unsigned int force\_error\_code)

Send a File to a client.

unsigned long SendErrorFile (struct AmmServer\_Instance \*instance, struct HTTPTransaction \*transaction, unsigned int errorCode)

Send an Error "file" response to a client , this is just a wrapper for a SendFile call with a force\_error\_code set.

 unsigned long SendMemoryBlockAsFile (char \*filename, int clientsock, char \*mem, unsigned long mem\_block)

Send a memory block to a client as a file.

## **Variables**

• unsigned int files open = 0

#### 6.39.1 Function Documentation

6.39.1.1 unsigned long SendErrorFile ( struct AmmServer\_Instance \* instance, struct HTTPTransaction \* transaction, unsigned int errorCode )

Send an Error "file" response to a client, this is just a wrapper for a SendFile call with a force\_error\_code set.

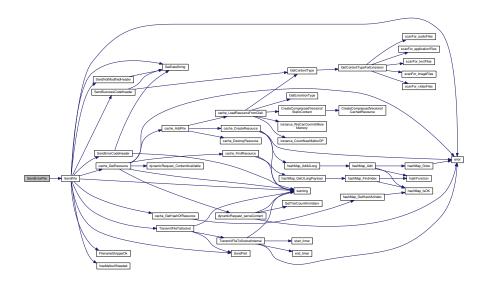
#### **Parameters**

An	AmmarServer Instance
HTTP-	this send file is part of
Transaction	
Error	Code to send

#### Return values

1=Success,0=Failure	

Here is the call graph for this function:



6.39.1.2 unsigned long SendFile ( struct AmmServer\_Instance \* instance, struct HTTPTransaction \* transaction, char \* verified\_filename\_pending\_copy, unsigned int force\_error\_code )

Send a File to a client.

#### **Parameters**

An	AmmarServer Instance
HTTP-	this send file is part of
Transaction	
Filename	that has been verified but has not been copied to the http checked for safety
Force	SendFile to fail with a specific error code (0 = dont force error)

#### Return values

1=Success,0=Failure	

Start sending the header first..! Due to error messages also having body payloads they are also handled here, creating clutter in the code but this way there is no need to write the same thing twice..!!

PRELIMINARY HEADER SENDING START -----

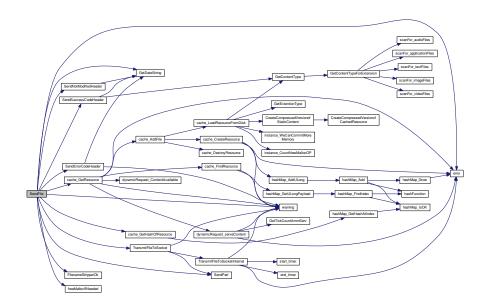
TODO Reorganize this: THIS SHOULD NOT BE SENT YET, SINCE WE MAY WANT TO EMMIT A 304 Not Modified Header if content is unmodified..!

PRELIMINARY HEADER SEND END -----

Serve cached file!

Serve file by reading it from disk!

Here is the call graph for this function:



6.39.1.3 unsigned long SendMemoryBlockAsFile ( char \* filename, int clientsock, char \* mem, unsigned long mem\_block )

Send a memory block to a client as a file.

## **Parameters**

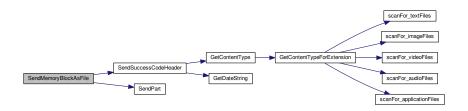
	Filename	to pretend that we are sending for
	Socket	we want to write to
Ì	Pointer	to memory that holds what we want to send to the client

Length	of memory block we want to send

Return values

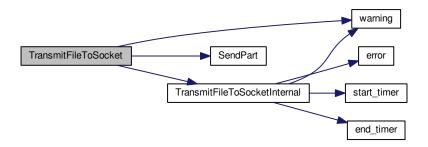
1=Success,0=Failure	

Here is the call graph for this function:



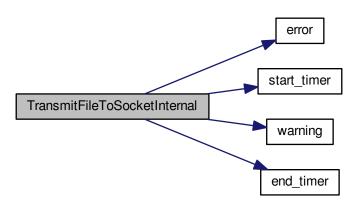
- 6.39.1.4 int SendPart ( int *clientsock*, char \* *message*, unsigned int *message\_size* )
- 6.39.1.5 int TransmitFileToSocket ( int *clientsock*, char \* *verified\_filename*, unsigned long *start\_at\_byte*, unsigned long *end\_at\_byte* )

Here is the call graph for this function:



6.39.1.6 int TransmitFileToSocketInternal ( FILE \* pFile, int clientsock, unsigned long bytesToSendStart ) [inline]

Here is the call graph for this function:



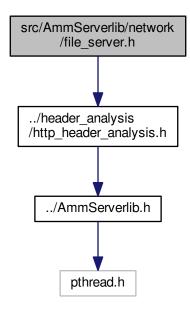
## 6.39.2 Variable Documentation

6.39.2.1 unsigned int files\_open = 0

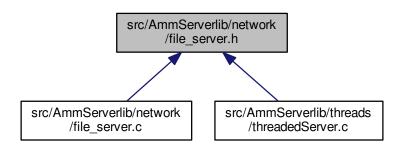
# 6.40 src/AmmServerlib/network/file\_server.h File Reference

Basic file server functionality of AmmarServer.

#include "../header\_analysis/http\_header\_analysis.h"
Include dependency graph for file\_server.h:



This graph shows which files directly or indirectly include this file:



## **Functions**

• unsigned long SendFile (struct AmmServer\_Instance \*instance, struct HTTPTransaction \*transaction, char \*verified\_filename\_pending\_copy, unsigned int force\_error\_code)

Send a File to a client.

• unsigned long SendErrorFile (struct AmmServer\_Instance \*instance, struct HTTPTransaction \*transaction, unsigned int errorCode)

Send an Error "file" response to a client, this is just a wrapper for a SendFile call with a force\_error\_code set.

 unsigned long SendMemoryBlockAsFile (char \*filename, int clientsock, char \*mem, unsigned long mem\_block)

Send a memory block to a client as a file.

## 6.40.1 Detailed Description

Basic file server functionality of AmmarServer.

**Author** 

Ammar Qammaz (AmmarkoV)

#### 6.40.2 Function Documentation

6.40.2.1 unsigned long SendErrorFile ( struct AmmServer\_Instance \* instance, struct HTTPTransaction \* transaction, unsigned int errorCode )

Send an Error "file" response to a client , this is just a wrapper for a SendFile call with a force error code set.

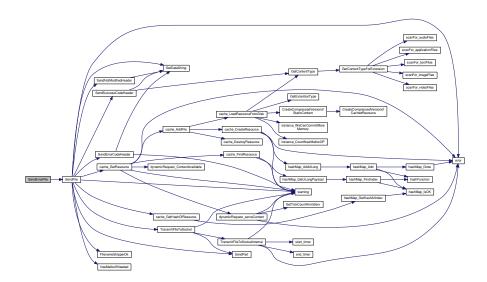
#### **Parameters**

An	AmmarServer Instance
HTTP-	this send file is part of
Transaction	
Error	Code to send

#### **Return values**

1=Success,0=Failure	

Here is the call graph for this function:



6.40.2.2 unsigned long SendFile ( struct AmmServer\_Instance \* instance, struct HTTPTransaction \* transaction, char \* verified\_filename\_pending\_copy, unsigned int force\_error\_code )

Send a File to a client.

#### **Parameters**

An	AmmarServer Instance
HTTP-	this send file is part of
Transaction	
Filename	that has been verified but has not been copied to the http checked for safety
Force	SendFile to fail with a specific error code (0 = dont force error)

#### **Return values**

1=Success,0=Failure	

Start sending the header first..! Due to error messages also having body payloads they are also handled here, creating clutter in the code but this way there is no need to write the same thing twice..!!

PRELIMINARY HEADER SENDING START -----

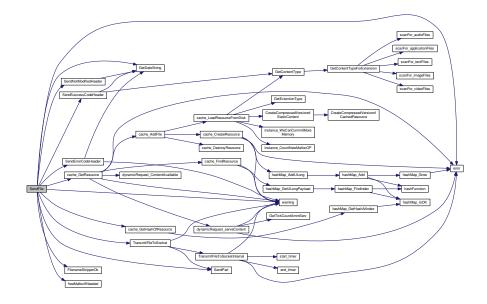
TODO Reorganize this: THIS SHOULD NOT BE SENT YET, SINCE WE MAY WANT TO EMMIT A 304 Not Modified Header if content is unmodified..!

PRELIMINARY HEADER SEND END -----

Serve cached file!

Serve file by reading it from disk!

Here is the call graph for this function:



6.40.2.3 unsigned long SendMemoryBlockAsFile ( char \* filename, int clientsock, char \* mem, unsigned long mem\_block )

Send a memory block to a client as a file.

## **Parameters**

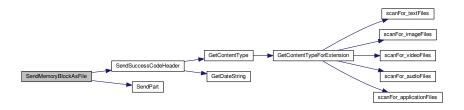
	Filename	to pretend that we are sending for
	Socket	we want to write to
Ì	Pointer	to memory that holds what we want to send to the client

Length	of memory block we want to send

**Return values** 

```
1=Success,0=Failure
```

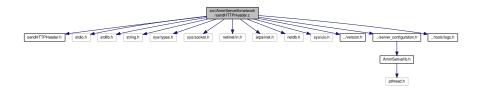
Here is the call graph for this function:



## 6.41 src/AmmServerlib/network/sendHTTPHeader.c File Reference

```
#include "sendHTTPHeader.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <sys/uio.h>
#include "../version.h"
#include "../server_configuration.h"
#include "../tools/logs.h"
```

Include dependency graph for sendHTTPHeader.c:



## **Functions**

 unsigned long SendErrorCodeHeader (int clientsock, unsigned int error\_code, char \*verified\_filename, char \*templates root)

Send an Error Code header.

- unsigned long SendSuccessCodeHeader (int clientsock, int success\_code, char \*verified\_filename)
  - Send a Success header, meaning that what was asked for will follow.
- unsigned long SendNotModifiedHeader (int clientsock)

Send a 304 Not Modified response.

• unsigned long SendAuthorizationHeader (int clientsock, char \*message, char \*verified\_filename)

Send a 401 Not Authorized response.

## 6.41.1 Function Documentation

6.41.1.1 unsigned long SendAuthorizationHeader ( int clientsock, char \* message,  $char * verified\_filename$  )

## Send a 401 Not Authorized response.

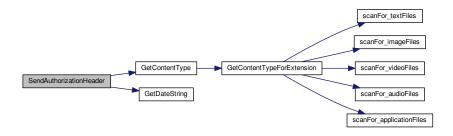
#### **Parameters**

Socket	to send to
String	with message to be sent
Verified	Filename of file asked to be transmitted

#### Return values

1=Success,0=Failure	
---------------------	--

Here is the call graph for this function:



6.41.1.2 unsigned long SendErrorCodeHeader ( int *clientsock*, unsigned int *error\_code*, char \* *verified\_filename*, char \* *templates\_root* )

## Send an Error Code header.

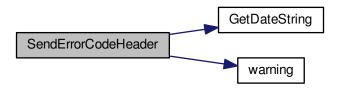
#### **Parameters**

Socket	to send to
ErrorCode	to be transmitted to client
Verified	Filename of file to transmit ( appended with error code )
Filename	to directory when error template files are stored

## Return values

1=Success,0=Failure
---------------------

Here is the call graph for this function:



## 6.41.1.3 unsigned long SendNotModifiedHeader ( int clientsock )

Send a 304 Not Modified response.

#### **Parameters**

Socket	to send to

#### Return values

```
1=Success,0=Failure
```

Here is the call graph for this function:



## 6.41.1.4 unsigned long SendSuccessCodeHeader ( int clientsock, int success\_code, char \* verified\_filename )

Send a Success header, meaning that what was asked for will follow.

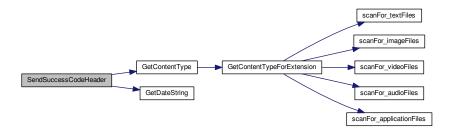
## **Parameters**

Socket	to send to
Success	code ( typically 200 ok )
Verified	Filename of file to transmit

## Return values

1=Success.0=Failure	

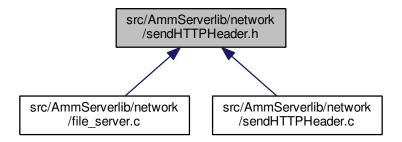
Here is the call graph for this function:



## 6.42 src/AmmServerlib/network/sendHTTPHeader.h File Reference

Small code segments that transmit HTTP responses.

This graph shows which files directly or indirectly include this file:



#### **Functions**

 unsigned long SendErrorCodeHeader (int clientsock, unsigned int error\_code, char \*verified\_filename, char \*templates\_root)

Send an Error Code header.

- unsigned long SendSuccessCodeHeader (int clientsock, int success\_code, char \*verified\_filename)

  Send a Success header, meaning that what was asked for will follow.
- unsigned long SendNotModifiedHeader (int clientsock)

Send a 304 Not Modified response.

• unsigned long SendAuthorizationHeader (int clientsock, char \*message, char \*verified\_filename)

Send a 401 Not Authorized response.

## 6.42.1 Detailed Description

Small code segments that transmit HTTP responses.

#### Author

Ammar Qammaz (AmmarkoV)

## 6.42.2 Function Documentation

6.42.2.1 unsigned long SendAuthorizationHeader ( int clientsock, char \* message, char \* verified\_filename )

## Send a 401 Not Authorized response.

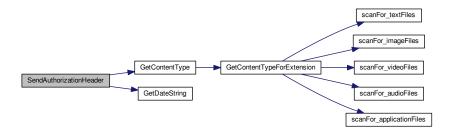
#### **Parameters**

Socket	to send to
String	with message to be sent
Verified	Filename of file asked to be transmitted

#### Return values

1=Success,0=Failure	

Here is the call graph for this function:



6.42.2.2 unsigned long SendErrorCodeHeader ( int *clientsock*, unsigned int *error\_code*, char \* *verified\_filename*, char \* *templates\_root* )

## Send an Error Code header.

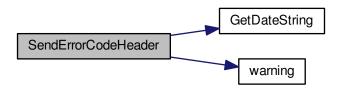
#### **Parameters**

Socket	to send to
ErrorCode	to be transmitted to client
Verified	Filename of file to transmit ( appended with error code )
Filename	to directory when error template files are stored

## Return values

1=Success,0=Failure
---------------------

Here is the call graph for this function:



## 6.42.2.3 unsigned long SendNotModifiedHeader ( int clientsock )

Send a 304 Not Modified response.

#### **Parameters**

0 1 1	
Socket	to send to
COONOL	to dona to

#### Return values

```
1=Success,0=Failure
```

Here is the call graph for this function:



6.42.2.4 unsigned long SendSuccessCodeHeader ( int clientsock, int success\_code, char \* verified\_filename )

Send a Success header, meaning that what was asked for will follow.

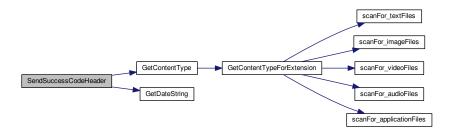
## Parameters

Socket	to send to
Success	code ( typically 200 ok )
Verified	Filename of file to transmit

## Return values

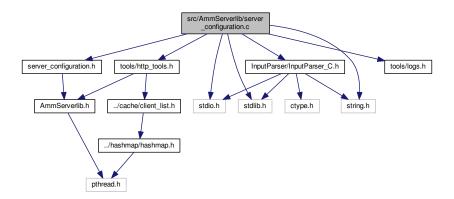
1=Success.0=Failure	

Here is the call graph for this function:



## 6.43 src/AmmServerlib/server\_configuration.c File Reference

```
#include "server_configuration.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "tools/http_tools.h"
#include "tools/logs.h"
#include "InputParser/InputParser_C.h"
Include dependency graph for server configuration.c:
```



#### **Functions**

 int instance\_WeCanCommitMoreMemory (struct AmmServer\_Instance \*instance, unsigned long additional-\_mem\_to\_malloc\_in\_bytes)

Check if we can commit more memory on an AmmarServer instance.

 int instance\_CountNewMallocOP (struct AmmServer\_Instance \*instance, unsigned long additional\_mem\_to-\_malloc\_in\_bytes)

Register a new memory Allocation to instance memory counters.

int instance\_CountFreeOP (struct AmmServer\_Instance \*instance, unsigned long additional\_mem\_to\_-malloc\_in\_bytes)

Register a new memory free operation to instance memory counters.

• int EmmitPossibleConfigurationWarnings (struct AmmServer\_Instance \*instance)

- int LoadConfigurationFile (struct AmmServer\_Instance \*instance, char \*conf\_file)
- int AssignStr (char \*\*dest, char \*source)
- int SetUsernameAndPassword (struct AmmServer Instance \*instance, char \*username, char \*password)

Set a username and password for clients to access specific webserver instance.

#### **Variables**

• unsigned int GLOBAL KILL SERVER SWITCH = 0

Setting this to 1 will signal that all instances of AmmarServer need to die at once.

char USERNAME\_UID\_FOR\_DAEMON [MAX\_FILE\_PATH] = DEFAULT\_USERNAME\_UID\_FOR\_DAEMON

Default Username that initially gets set to DEFAULT\_USERNAME\_UID\_FOR\_DAEMON but can be changed through a configuration file.

- int CHANGE\_TO\_UID =NON\_ROOT\_UID\_IF\_USER\_FAILS
- signed int CHANGE\_PRIORITY =0

Value that gets set from configuration files , and if it is non-zero it will trigger a priority change ( change nice value )

- int varSocketTimeoutREAD\_seconds = DEFAULT\_SOCKET\_READ\_TIMEOUT\_SECS
- int varSocketTimeoutWRITE\_seconds =DEFAULT\_SOCKET\_WRITE\_TIMEOUT\_SECS
- unsigned char CACHING\_ENABLED =1

If caching is disabled server becomes a very simple file server, dynamic requests are also disabled.

int MAX\_SEPERATE\_CACHE\_ITEMS = 1024

Maximum Number of separate items in cache ( per instance of AmmarServer )

• int MAX\_CACHE\_SIZE\_IN\_MB = 128

Maximum memory usage (Megabytes) for the entire cache (per instance of AmmarServer)

int MAX\_CACHE\_SIZE\_FOR\_EACH\_FILE\_IN\_MB = 3

Maximum memory usage ( Megabytes ) for a specific entry of the cache ( per instance of AmmarServer )

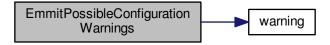
- int AccessLogEnable =0
- char AccessLog [MAX\_FILE\_PATH] = "access.log"
- int ErrorLogEnable =0
- char ErrorLog [MAX FILE PATH] ="error.log"
- char TemplatesInternalURI [MAX RESOURCE] = TEMPLATE INTERNAL URI

#### 6.43.1 Function Documentation

```
6.43.1.1 int AssignStr ( char ** dest, char * source )
```

6.43.1.2 int EmmitPossibleConfigurationWarnings ( struct AmmServer\_Instance \* instance )

Here is the call graph for this function:



6.43.1.3 int instance\_CountFreeOP ( struct AmmServer\_Instance \* instance, unsigned long additional\_mem\_to\_malloc\_in\_bytes )

Register a new memory free operation to instance memory counters.

#### **Parameters**

An AmmarServer instance	
Memory	that was freed

#### Return values

1=Success,0=Failure	

6.43.1.4 int instance\_CountNewMallocOP ( struct AmmServer\_Instance \* instance, unsigned long additional\_mem\_to\_malloc\_in\_bytes )

Register a new memory Allocation to instance memory counters.

#### **Parameters**

An	AmmarServer instance
Memory	that was additionally allocated

#### **Return values**

1=Success,0=Failure	

6.43.1.5 int instance\_WeCanCommitMoreMemory ( struct AmmServer\_Instance \* instance, unsigned long additional\_mem\_to\_malloc\_in\_bytes )

Check if we can commit more memory on an AmmarServer instance.

#### **Parameters**

An	AmmarServer instance
Memory	to additionally allocate

## Return values

1=Ok,0=Don'tAllocate	

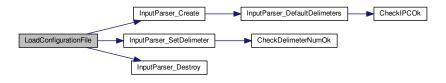
6.43.1.6 int LoadConfigurationFile ( struct AmmServer\_Instance \* instance, char \* conf\_file )

#### **Parameters**

Load a configuration file
---------------------------

**Bug** LoadConfigurationFiles etc is not ready yet, although it relies on InputParser and should be easy to implement, there are just things missing still and that's why I postpone implementing it

Here is the call graph for this function:



6.43.1.7 int SetUsernameAndPassword ( struct AmmServer\_Instance \* instance, char \* username, char \* password )

Set a username and password for clients to access specific webserver instance.

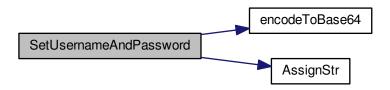
#### **Parameters**

An	AmmarServer instance
String	with new username
String	with new password

#### Return values

1=Success,0=Failure	

Here is the call graph for this function:



## 6.43.2 Variable Documentation

6.43.2.1 char AccessLog[MAX\_FILE\_PATH] ="access.log"

6.43.2.2 int AccessLogEnable =0

6.43.2.3 unsigned char CACHING\_ENABLED =1

If caching is disabled server becomes a very simple file server, dynamic requests are also disabled.

6.43.2.4 signed int CHANGE\_PRIORITY =0

Value that gets set from configuration files, and if it is non-zero it will trigger a priority change (change nice value)

6.43.2.5 int CHANGE\_TO\_UID = NON\_ROOT\_UID\_IF\_USER\_FAILS

6.43.2.6 char ErrorLog[MAX\_FILE\_PATH] ="error.log"

6.43.2.7 int ErrorLogEnable =0

6.43.2.8 unsigned int GLOBAL\_KILL\_SERVER\_SWITCH = 0

Setting this to 1 will signal that all instances of AmmarServer need to die at once.

6.43.2.9 int MAX\_CACHE\_SIZE\_FOR\_EACH\_FILE\_IN\_MB = 3

Maximum memory usage ( Megabytes ) for a specific entry of the cache ( per instance of AmmarServer )

6.43.2.10 int MAX\_CACHE\_SIZE\_IN\_MB = 128

Maximum memory usage ( Megabytes ) for the entire cache ( per instance of AmmarServer )

6.43.2.11 int MAX\_SEPERATE\_CACHE\_ITEMS = 1024

Maximum Number of separate items in cache ( per instance of AmmarServer )

- 6.43.2.12 char TemplatesInternalURI[MAX\_RESOURCE] = TEMPLATE\_INTERNAL\_URI
- 6.43.2.13 char USERNAME\_UID\_FOR\_DAEMON[MAX\_FILE\_PATH] = DEFAULT\_USERNAME\_UID\_FOR\_DAEMON

Default Username that initially gets set to DEFAULT\_USERNAME\_UID\_FOR\_DAEMON but can be changed through a configuration file.

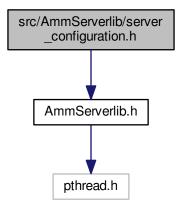
- 6.43.2.14 int varSocketTimeoutREAD\_seconds = DEFAULT\_SOCKET\_READ\_TIMEOUT\_SECS
- 6.43.2.15 int varSocketTimeoutWRITE\_seconds = DEFAULT\_SOCKET\_WRITE\_TIMEOUT\_SECS

# 6.44 src/AmmServerlib/server\_configuration.h File Reference

The Main Header for the settings used by AmmarServer.

#include "AmmServerlib.h"

Include dependency graph for server\_configuration.h:



This graph shows which files directly or indirectly include this file:



#### **Macros**

#define THREAD\_SLEEP\_TIME\_WHEN\_OUR\_PRESPAWNED\_THREAD\_IS\_NEXT 700

Next prespawned thread , should be vigilant and ready to serve so it has a shorter delay than the other prespawned threads ( 0.7ms max delay seems like a good value )

#define THREAD\_SLEEP\_TIME\_FOR\_PRESPAWNED\_THREADS 25000

Sleep time for threads that are prespawned until they check for potential new work, the lowest the value here, the shortest the wait time for clients, but this causes higher CPU usage (for idle tasks) and ultimately more power consumption A good default time is 25000, (25ms)

• #define CALCULATE\_TIME\_FOR\_UPLOADS 1

Calculate (And output) transmission speed for files broadcast by AmmarServer.

#define COMPILE\_WITH\_CLIENT\_LIST 0

Precompiler switch that controls baking in ( or not ) the client list capabilities , currently disabled since client lists are not yet implemented.

#define MAX\_CLIENTS\_LISTENING\_FOR 5000

Maximum Target of concurrent clients being listened at the same time C10K tests require this to be 10000 (http-://en.wikipedia.org/wiki/C10k\_problem)

• #define MAX CLIENT THREADS 3000

Maximum Number of concurrent threads being created at the same time, depending on the size of the listen pool this can be smaller than the MAX\_CLIENTS\_LISTENING\_FOR and connections will be queued and served sequentially.

#define MAX\_CLIENT\_PRESPAWNED\_THREADS 0

Prespawned theads reduce overall latency but they increase CPU load, 0 disables them.

#define MAX CLIENTS PER IP 3

Maximum connections per IP, this is a little dangerous since multiple PC's can have a single gateway, but it is a good heuristic to better share resources.

#define MAX\_HTTP\_REQUEST\_HEADER\_LINES 1024

An incoming header should not have more than X numbers of lines.

• #define MAX RESOURCE SLASHES 15

Max slashes in a Resource (i.e. http://xxx.xxx.xxx/test/resource has 4 slashes.

• #define MAX CONFIGURATION FILE LINE SIZE 512

Maximum line length in configuration file.

• #define MAX CONTENT TYPE 128

Maximum length of a content type record.

#define MAX\_FILE\_READ\_BLOCK\_KB 1024

Length of blocks allocated, read and sent in order to transmit a file to a client, bigger values read faster from the disk and possibly better utilize bandwidth in the expense of memory consumption.

• #define MAX\_HTTP\_REQUEST\_HEADER 4096

Maximum size of an incoming HTTP Header.

#define HTTP\_POST\_GROWTH\_STEP\_REQUEST\_HEADER 512/\*KB\*/\*1024

Maximum size of an incoming HTTP Header allocation step.

• #define MAX HTTP POST REQUEST HEADER 4/\*MB\*/\*1024\*1024

Maximum size of an incoming POST Header , since it carries files this should be big enough ( say 4 MB )

• #define MAX\_HTTP\_REQUEST\_HEADER\_REPLY 1024

Maximum size of an http header reply.

#define INITIAL DIRECTORY LIST RESPONSE BODY 64/\*KB\*/\*1024

Controls initial allocated size for a directory listing.

#define GROWSTEP\_DIRECTORY\_LIST\_RESPONSE\_BODY 16/\*KB\*/\*1024

Controls allocation step for when we run out of space for a directory listing.

• #define MAX\_DIRECTORY\_LIST\_RESPONSE\_BODY 128/\*KB\*/\*1024

Maximum space allocated for a directory listing.

• #define REALLOC TO SAVE MORE THAN THIS NUMBER BYTES 4096

When we compress a file we may have a buffer allocated for 16KB and the compressed size might be 1.6KB (if we get an impressive 1:10 ratio) If that's the case we could do a system call to free memory and allocate a 1.6KB chunk of memory thus being economic in memory requirements.

#define ENABLE AUTOMATIC CONFIGURATION LOADING 1

If this enabled and we haven't specified a configuration file we will try to open an ammarServer.conf.

#define ENABLE\_POST 1

Enable POST request handling, switching this to 0 will completely deny them reducing attack surface.

#define ENABLE COMPRESSION 0

Enable Compression using ZLib, this increases CPU usage, code surface, requires the zlib library to be linked, but on the other hand conserves bandwidth and memory.

• #define ENABLE DYNAMIC CONTENT COMPRESSION 0

Enable Compression for dynamic content, this can be tuned per dynamic resource, but this is a global switch for all nodes This generally doesnt seem like a very good idea unless you have a dynamic html file of 20KB+ with very rare changes to compensate for the overhead.

#define ENABLE\_DROPPING\_ROOT\_UID\_IF\_ROOT 1

In order to bind ports under 1000, a process needs to have Super user UID, after we bind the port we really don't want to have our process running as a super user, it is a serious security liability This should always be 1.

#define ENABLE DROPPING UID ALWAYS 0

If this is enabled we will always change our UID no matter if we are a super user or not ( if this is disabled only super user processes will get the UID change )

#define DEFAULT\_USERNAME\_UID\_FOR\_DAEMON "www-data"

Default Username to change to if we are running from root.

#define NON ROOT UID IF USER FAILS 1500

Non Root UID to change to.

#define ENABLE\_INTERNAL\_RESOURCES\_RESOLVE 1

Resolve internal resources to redirect them to point templates (this should always be 1, although its implementation is a little dodgy right now)

#define ENABLE\_DIRECTORY\_LISTING 1

Enable directory listing, if this is disabled attack surface gets significantly reduced.

• #define EPOCH YEAR IN TM YEAR 1900

TM structures carry the year after 1900 (see http://www.cplusplus.com/reference/ctime/tm/) so this is encoded here as a reminder.

#define DEFAULT SOCKET READ TIMEOUT SECS 4

Default timeout value before which a socket blocking on a read call should be considered dead.

• #define DEFAULT SOCKET WRITE TIMEOUT SECS 4

Default timeout value before which a socket blocking on a write call should be considered dead.

#define TEMPLATE\_INTERNAL\_URI "\_asvres\_/"

String that corresponds to the template directory ( for directory\_lists )

## **Functions**

int instance\_WeCanCommitMoreMemory (struct AmmServer\_Instance \*instance, unsigned long additional\_mem\_to\_malloc\_in\_bytes)

Check if we can commit more memory on an AmmarServer instance.

 int instance\_CountNewMallocOP (struct AmmServer\_Instance \*instance, unsigned long additional\_mem\_to-\_malloc\_in\_bytes)

Register a new memory Allocation to instance memory counters.

int instance\_CountFreeOP (struct AmmServer\_Instance \*instance, unsigned long additional\_mem\_to\_-malloc\_in\_bytes)

Register a new memory free operation to instance memory counters.

int EmmitPossibleConfigurationWarnings ()

Internal check of server configuration and possible error messages in impossible situations.

int LoadConfigurationFile (struct AmmServer\_Instance \*instance, char \*conf\_file)

- int AssignStr (char \*\*dest, char \*source)
- int SetUsernameAndPassword (struct AmmServer\_Instance \*instance, char \*username, char \*password)

Set a username and password for clients to access specific webserver instance.

#### **Variables**

unsigned int GLOBAL\_KILL\_SERVER\_SWITCH

Setting this to 1 will signal that all instances of AmmarServer need to die at once.

char USERNAME\_UID\_FOR\_DAEMON [MAX\_FILE\_PATH]

Default Username that initially gets set to DEFAULT\_USERNAME\_UID\_FOR\_DAEMON but can be changed through a configuration file.

- int CHANGE\_TO\_UID
- int CHANGE\_PRIORITY

Value that gets set from configuration files , and if it is non-zero it will trigger a priority change ( change nice value )

- int varSocketTimeoutREAD seconds
- int varSocketTimeoutWRITE seconds
- unsigned char CACHING\_ENABLED

If caching is disabled server becomes a very simple file server, dynamic requests are also disabled.

• int MAX SEPERATE CACHE ITEMS

Maximum Number of separate items in cache ( per instance of AmmarServer )

int MAX\_CACHE\_SIZE\_IN\_MB

Maximum memory usage ( Megabytes ) for the entire cache ( per instance of AmmarServer )

· int MAX CACHE SIZE FOR EACH FILE IN MB

Maximum memory usage ( Megabytes ) for a specific entry of the cache ( per instance of AmmarServer )

- int AccessLogEnable
- char AccessLog [MAX FILE PATH]
- int ErrorLogEnable
- char ErrorLog [MAX\_FILE\_PATH]
- char TemplatesInternalURI [MAX\_RESOURCE]

## 6.44.1 Detailed Description

The Main Header for the settings used by AmmarServer. Take extra care when changing something here , since its impact is global

**Author** 

Ammar Qammaz (AmmarkoV)

Bug Server configuration at some point should be ported from defines to a per instance configuration file, some of these defines will always remain since they control global allocations

#### 6.44.2 Macro Definition Documentation

6.44.2.1 #define CALCULATE\_TIME\_FOR\_UPLOADS 1

Calculate (And output) transmission speed for files broadcast by AmmarServer.

6.44.2.2 #define COMPILE\_WITH\_CLIENT\_LIST 0

Precompiler switch that controls baking in ( or not ) the client list capabilities, currently disabled since client lists are not yet implemented.

#### 6.44.2.3 #define DEFAULT\_SOCKET\_READ\_TIMEOUT\_SECS 4

Default timeout value before which a socket blocking on a read call should be considered dead.

#### 6.44.2.4 #define DEFAULT\_SOCKET\_WRITE\_TIMEOUT\_SECS 4

Default timeout value before which a socket blocking on a write call should be considered dead.

#### 6.44.2.5 #define DEFAULT\_USERNAME\_UID\_FOR\_DAEMON "www-data"

Default Username to change to if we are running from root.

#### 6.44.2.6 #define ENABLE\_AUTOMATIC\_CONFIGURATION\_LOADING 1

If this enabled and we haven't specified a configuration file we will try to open an ammarServer.conf.

#### 6.44.2.7 #define ENABLE COMPRESSION 0

Enable Compression using ZLib , this increases CPU usage , code surface , requires the zlib library to be linked , but on the other hand conserves bandwidth and memory.

#### 6.44.2.8 #define ENABLE\_DIRECTORY\_LISTING 1

Enable directory listing, if this is disabled attack surface gets significantly reduced.

## 6.44.2.9 #define ENABLE\_DROPPING\_ROOT\_UID\_IF\_ROOT 1

In order to bind ports under 1000, a process needs to have Super user UID, after we bind the port we *really* don't want to have our process running as a super user, it is a serious security liability This should always be 1.

#### 6.44.2.10 #define ENABLE\_DROPPING\_UID\_ALWAYS 0

If this is enabled we will always change our UID no matter if we are a super user or not ( if this is disabled only super user processes will get the UID change )

#### 6.44.2.11 #define ENABLE\_DYNAMIC\_CONTENT\_COMPRESSION 0

Enable Compression for dynamic content , this can be tuned per dynamic resource , but this is a global switch for all nodes This generally doesnt seem like a very good idea unless you have a dynamic html file of 20KB+ with very rare changes to compensate for the overhead.

## 6.44.2.12 #define ENABLE\_INTERNAL\_RESOURCES\_RESOLVE 1

Resolve internal resources to redirect them to point templates ( this should always be 1, although its implementation is a little dodgy right now )

#### 6.44.2.13 #define ENABLE\_POST 1

Enable POST request handling, switching this to 0 will completely deny them reducing attack surface.

6.44.2.14 #define EPOCH\_YEAR\_IN\_TM\_YEAR 1900

TM structures carry the year after 1900 (see http://www.cplusplus.com/reference/ctime/tm/) so this is encoded here as a reminder.

6.44.2.15 #define GROWSTEP\_DIRECTORY\_LIST\_RESPONSE\_BODY 16/\*KB\*/\*1024

Controls allocation step for when we run out of space for a directory listing.

6.44.2.16 #define HTTP\_POST\_GROWTH\_STEP\_REQUEST\_HEADER 512/\*KB\*/\*1024

Maximum size of an incoming HTTP Header allocation step.

6.44.2.17 #define INITIAL\_DIRECTORY\_LIST\_RESPONSE\_BODY 64/\*KB\*/\*1024

Controls initial allocated size for a directory listing.

6.44.2.18 #define MAX\_CLIENT\_PRESPAWNED\_THREADS 0

Prespawned theads reduce overall latency but they increase CPU load, 0 disables them.

6.44.2.19 #define MAX\_CLIENT\_THREADS 3000

Maximum Number of concurrent threads being created at the same time , depending on the size of the listen pool this can be smaller than the MAX\_CLIENTS\_LISTENING\_FOR and connections will be queued and served sequentially.

6.44.2.20 #define MAX\_CLIENTS\_LISTENING\_FOR 5000

Maximum Target of concurrent clients being listened at the same time C10K tests require this to be 10000 (http-://en.wikipedia.org/wiki/C10k\_problem)

6.44.2.21 #define MAX\_CLIENTS\_PER\_IP 3

Maximum connections per IP , this is a little dangerous since multiple PC's can have a single gateway , but it is a good heuristic to better share resources.

Bug MAX\_CLIENTS\_PER\_IP is not used if there is no client list declared

6.44.2.22 #define MAX\_CONFIGURATION\_FILE\_LINE\_SIZE 512

Maximum line length in configuration file.

6.44.2.23 #define MAX\_CONTENT\_TYPE 128

Maximum length of a content type record.

6.44.2.24 #define MAX\_DIRECTORY\_LIST\_RESPONSE\_BODY 128/\*KB\*/\*1024

Maximum space allocated for a directory listing.

6.44.2.25 #define MAX\_FILE\_READ\_BLOCK\_KB 1024

Length of blocks allocated, read and sent in order to transmit a file to a client, bigger values read faster from the disk and possibly better utilize bandwidth in the expense of memory consumption.

6.44.2.26 #define MAX\_HTTP\_POST\_REQUEST\_HEADER 4/\*MB\*/\*1024\*1024

Maximum size of an incoming POST Header, since it carries files this should be big enough ( say 4 MB)

6.44.2.27 #define MAX\_HTTP\_REQUEST\_HEADER 4096

Maximum size of an incoming HTTP Header.

6.44.2.28 #define MAX\_HTTP\_REQUEST\_HEADER\_LINES 1024

An incoming header should not have more than X numbers of lines.

6.44.2.29 #define MAX\_HTTP\_REQUEST\_HEADER\_REPLY 1024

Maximum size of an http header reply.

6.44.2.30 #define MAX\_RESOURCE\_SLASHES 15

Max slashes in a Resource (i.e. http://xxx.xxx.xxx/test/resource has 4 slashes.

6.44.2.31 #define NON\_ROOT\_UID\_IF\_USER\_FAILS 1500

Non Root UID to change to.

6.44.2.32 #define REALLOC\_TO\_SAVE\_MORE\_THAN\_THIS\_NUMBER\_BYTES 4096

When we compress a file we may have a buffer allocated for 16KB and the compressed size might be 1.6KB ( if we get an impressive 1:10 ratio ) If that's the case we could do a system call to free memory and allocate a 1.6KB chunk of memory thus being economic in memory requirements.

6.44.2.33 #define TEMPLATE\_INTERNAL\_URI "\_asvres\_/"

String that corresponds to the template directory (for directory lists)

Bug Please note that the file server has limits for filenames so this should not be very long *asvres*/filename.jpg is OK a filename like *asvres*/filenamemplampla.jpg will return a 404

6.44.2.34 #define THREAD\_SLEEP\_TIME\_FOR\_PRESPAWNED\_THREADS 25000

Sleep time for threads that are prespawned until they check for potential new work , the lowest the value here , the shortest the wait time for clients , but this causes higher CPU usage ( for idle tasks ) and ultimately more power consumption A good default time is 25000, (25ms)

6.44.2.35 #define THREAD\_SLEEP\_TIME\_WHEN\_OUR\_PRESPAWNED\_THREAD\_IS\_NEXT 700

Next prespawned thread , should be vigilant and ready to serve so it has a shorter delay than the other prespawned threads (0.7ms max delay seems like a good value)

#### 6.44.3 Function Documentation

```
6.44.3.1 int AssignStr ( char ** dest, char * source )
```

6.44.3.2 int EmmitPossibleConfigurationWarnings ( )

Internal check of server configuration and possible error messages in impossible situations.

#### Return values

```
1=Success,0=Failure
```

6.44.3.3 int instance\_CountFreeOP ( struct AmmServer\_Instance \* instance, unsigned long additional\_mem\_to\_malloc\_in\_bytes )

Register a new memory free operation to instance memory counters.

#### **Parameters**

An	AmmarServer instance
Memory	that was freed

#### Return values

1=Success,0=Failure	
---------------------	--

6.44.3.4 int instance\_CountNewMallocOP ( struct AmmServer\_Instance \* instance, unsigned long additional\_mem\_to\_malloc\_in\_bytes )

Register a new memory Allocation to instance memory counters.

## **Parameters**

An	AmmarServer instance
Memory	that was additionally allocated

#### Return values

1=Success,0=Failure	

6.44.3.5 int instance\_WeCanCommitMoreMemory ( struct AmmServer\_Instance \* instance, unsigned long additional\_mem\_to\_malloc\_in\_bytes )

Check if we can commit more memory on an AmmarServer instance.

#### **Parameters**

An	AmmarServer instance

Memory	to additionally allocate
Return values	

1-0k	0=Don	'tAllocate
I - OR	U-DUI1	inilocate

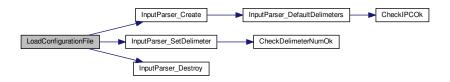
6.44.3.6 int LoadConfigurationFile ( struct AmmServer\_Instance \* instance, char \* conf\_file )

#### **Parameters**

Load	d a configuration file	

**Bug** LoadConfigurationFiles etc is not ready yet, although it relies on InputParser and should be easy to implement, there are just things missing still and that's why I postpone implementing it

Here is the call graph for this function:



6.44.3.7 int SetUsernameAndPassword ( struct AmmServer\_Instance \* instance, char \* username, char \* password )

Set a username and password for clients to access specific webserver instance.

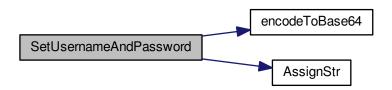
## **Parameters**

An	AmmarServer instance
String	with new username
String	with new password

#### **Return values**

1=Success,0=Failure	

Here is the call graph for this function:



6.44.4 Variable Documentation

6.44.4.1 char AccessLog[MAX\_FILE\_PATH]

6.44.4.2 int AccessLogEnable

6.44.4.3 unsigned char CACHING\_ENABLED

If caching is disabled server becomes a very simple file server, dynamic requests are also disabled.

6.44.4.4 int CHANGE\_PRIORITY

Value that gets set from configuration files, and if it is non-zero it will trigger a priority change (change nice value)

6.44.4.5 int CHANGE\_TO\_UID

6.44.4.6 char ErrorLog[MAX\_FILE\_PATH]

6.44.4.7 int ErrorLogEnable

6.44.4.8 unsigned int GLOBAL\_KILL\_SERVER\_SWITCH

Setting this to 1 will signal that all instances of AmmarServer need to die at once.

6.44.4.9 int MAX\_CACHE\_SIZE\_FOR\_EACH\_FILE\_IN\_MB

Maximum memory usage ( Megabytes ) for a specific entry of the cache ( per instance of AmmarServer )

6.44.4.10 int MAX\_CACHE\_SIZE\_IN\_MB

Maximum memory usage ( Megabytes ) for the entire cache ( per instance of AmmarServer )

6.44.4.11 int MAX\_SEPERATE\_CACHE\_ITEMS

Maximum Number of separate items in cache (per instance of AmmarServer)

6.44.4.12 char TemplatesInternalURI[MAX RESOURCE]

6.44.4.13 char USERNAME\_UID\_FOR\_DAEMON[MAX\_FILE\_PATH]

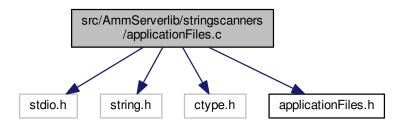
Default Username that initially gets set to DEFAULT\_USERNAME\_UID\_FOR\_DAEMON but can be changed through a configuration file.

6.44.4.14 int varSocketTimeoutREAD\_seconds

6.44.4.15 int varSocketTimeoutWRITE\_seconds

# 6.45 src/AmmServerlib/stringscanners/applicationFiles.c File Reference

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>
#include "applicationFiles.h"
Include dependency graph for applicationFiles.c:
```



## **Functions**

int scanFor\_applicationFiles (char \*str, unsigned int strLength)

Scan a string for one of the words of the applicationFiles word set.

#### 6.45.1 Function Documentation

6.45.1.1 int scanFor\_applicationFiles ( char \* str, unsigned int strLength )

Scan a string for one of the words of the applicationFiles word set.

#### **Parameters**

Input	String , to be scanned
Length	of Input String

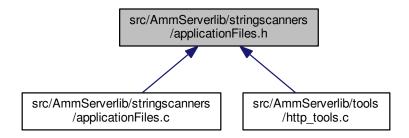
#### Return values

See above enumerator
----------------------

# 6.46 src/AmmServerlib/stringscanners/applicationFiles.h File Reference

A tool that scans for a string in a very fast and robust way.

This graph shows which files directly or indirectly include this file:



#### **Enumerations**

enum {
 APPLICATIONFILES\_EMPTY =0, APPLICATIONFILES\_EXE, APPLICATIONFILES\_DLL, APPLICATIONFILES\_SCR,
 APPLICATIONFILES\_CPL, APPLICATIONFILES\_SWF, APPLICATIONFILES\_PDF, APPLICATIONFILES\_END\_OF\_ITEMS }

Enumerator for the IDs of applicationFiles so we can know what the result was.

## **Functions**

int scanFor\_applicationFiles (char \*str, unsigned int strLength)
 Scan a string for one of the words of the applicationFiles word set.

## 6.46.1 Detailed Description

A tool that scans for a string in a very fast and robust way.

Author

Ammar Qammaz (AmmarkoV)

## 6.46.2 Enumeration Type Documentation

6.46.2.1 anonymous enum

Enumerator for the IDs of applicationFiles so we can know what the result was.

#### **Enumerator**

APPLICATIONFILES\_EMPTY
APPLICATIONFILES\_EXE
APPLICATIONFILES\_DLL
APPLICATIONFILES\_SCR
APPLICATIONFILES\_CPL
APPLICATIONFILES\_SWF

# APPLICATIONFILES\_PDF APPLICATIONFILES\_END\_OF\_ITEMS

## 6.46.3 Function Documentation

6.46.3.1 int scanFor\_applicationFiles ( char \* str, unsigned int strLength )

Scan a string for one of the words of the applicationFiles word set.

#### **Parameters**

Input	String, to be scanned
Length	of Input String

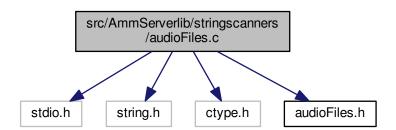
#### Return values

See	above enumerator

# 6.47 src/AmmServerlib/stringscanners/audioFiles.c File Reference

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>
#include "audioFiles.h"
```

Include dependency graph for audioFiles.c:



## **Functions**

int scanFor\_audioFiles (char \*str, unsigned int strLength)
 Scan a string for one of the words of the audioFiles word set.

## 6.47.1 Function Documentation

6.47.1.1 int scanFor\_audioFiles ( char \* str, unsigned int strLength )

Scan a string for one of the words of the audioFiles word set.

#### **Parameters**

Input	nput String, to be scanned	
Length	of Input String	

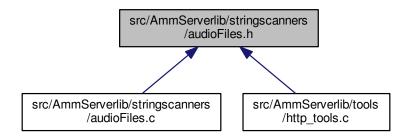
#### Return values

See	above enumerator

# 6.48 src/AmmServerlib/stringscanners/audioFiles.h File Reference

A tool that scans for a string in a very fast and robust way.

This graph shows which files directly or indirectly include this file:



## **Enumerations**

enum {
 AUDIOFILES\_EMPTY =0, AUDIOFILES\_MP3, AUDIOFILES\_WAV, AUDIOFILES\_MID,
 AUDIOFILES\_OGG, AUDIOFILES\_VOC, AUDIOFILES\_AU, AUDIOFILES\_END\_OF\_ITEMS }

Enumerator for the IDs of audioFiles so we can know what the result was.

## **Functions**

int scanFor\_audioFiles (char \*str, unsigned int strLength)
 Scan a string for one of the words of the audioFiles word set.

## 6.48.1 Detailed Description

A tool that scans for a string in a very fast and robust way.

#### **Author**

Ammar Qammaz (AmmarkoV)

## 6.48.2 Enumeration Type Documentation

## 6.48.2.1 anonymous enum

Enumerator for the IDs of audioFiles so we can know what the result was.

#### Enumerator

**AUDIOFILES\_EMPTY** 

AUDIOFILES\_MP3

AUDIOFILES\_WAV

AUDIOFILES\_MID

AUDIOFILES\_OGG

AUDIOFILES\_VOC

AUDIOFILES\_AU

AUDIOFILES\_END\_OF\_ITEMS

#### 6.48.3 Function Documentation

6.48.3.1 int scanFor\_audioFiles ( char \* str, unsigned int strLength )

Scan a string for one of the words of the audioFiles word set.

#### **Parameters**

Input	String , to be scanned
Length	of Input String

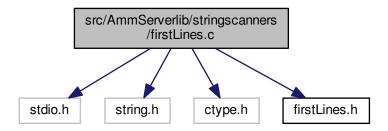
## Return values

See	above enumerator
-----	------------------

# 6.49 src/AmmServerlib/stringscanners/firstLines.c File Reference

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>
#include "firstLines.h"
```

Include dependency graph for firstLines.c:



## **Functions**

int scanFor\_firstLines (char \*str, unsigned int strLength)
 Scan a string for one of the words of the firstLines word set.

## 6.49.1 Function Documentation

6.49.1.1 int scanFor\_firstLines ( char \* str, unsigned int strLength )

Scan a string for one of the words of the firstLines word set.

#### **Parameters**

Input	String , to be scanned
Length	of Input String

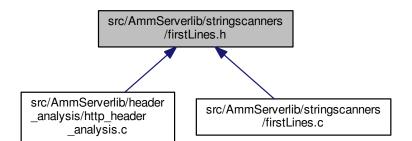
#### Return values

See	above enumerator

# 6.50 src/AmmServerlib/stringscanners/firstLines.h File Reference

A tool that scans for a string in a very fast and robust way.

This graph shows which files directly or indirectly include this file:



## **Enumerations**

enum {

FIRSTLINES\_EMPTY =0, FIRSTLINES\_GET, FIRSTLINES\_HEAD, FIRSTLINES\_POST, FIRSTLINES\_PUT, FIRSTLINES\_DELETE, FIRSTLINES\_TRACE, FIRSTLINES\_OPTIONS, FIRSTLINES\_CONNECT, FIRSTLINES\_PATCH, FIRSTLINES\_END\_OF\_ITEMS }

Enumerator for the IDs of firstLines so we can know what the result was.

## **Functions**

int scanFor\_firstLines (char \*str, unsigned int strLength)
 Scan a string for one of the words of the firstLines word set.

## 6.50.1 Detailed Description

A tool that scans for a string in a very fast and robust way.

#### **Author**

Ammar Qammaz (AmmarkoV)

## 6.50.2 Enumeration Type Documentation

6.50.2.1 anonymous enum

Enumerator for the IDs of firstLines so we can know what the result was.

#### **Enumerator**

FIRSTLINES\_EMPTY

FIRSTLINES\_GET

FIRSTLINES\_HEAD

FIRSTLINES\_POST

FIRSTLINES\_PUT

FIRSTLINES\_DELETE

FIRSTLINES\_TRACE

FIRSTLINES\_OPTIONS

FIRSTLINES\_CONNECT

FIRSTLINES\_PATCH

FIRSTLINES\_END\_OF\_ITEMS

## 6.50.3 Function Documentation

6.50.3.1 int scanFor\_firstLines ( char \* str, unsigned int strLength )

Scan a string for one of the words of the firstLines word set.

## **Parameters**

Input	String, to be scanned
Length	of Input String

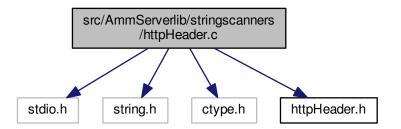
#### **Return values**

See	above enumerator

# 6.51 src/AmmServerlib/stringscanners/httpHeader.c File Reference

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>
#include "httpHeader.h"
```

Include dependency graph for httpHeader.c:



## **Functions**

int scanFor\_httpHeader (char \*str, unsigned int strLength)

Scan a string for one of the words of the httpHeader word set.

### 6.51.1 Function Documentation

6.51.1.1 int scanFor\_httpHeader ( char \* str, unsigned int strLength )

Scan a string for one of the words of the httpHeader word set.

## Parameters

Input	String, to be scanned
Length	of Input String

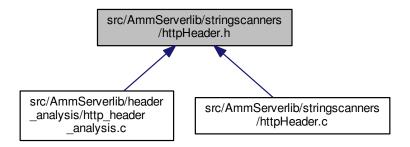
### Return values

See	above enumerator

# 6.52 src/AmmServerlib/stringscanners/httpHeader.h File Reference

A tool that scans for a string in a very fast and robust way.

This graph shows which files directly or indirectly include this file:



### **Enumerations**

• enum {

HTTPHEADER\_EMPTY =0, HTTPHEADER\_AUTHORIZATION, HTTPHEADER\_ACCEPT\_ENCODING, HTTPHEADER COOKIE,

HTTPHEADER\_CONNECTION, HTTPHEADER\_HOST, HTTPHEADER\_IF\_NONE\_MATCH, HTTPHEADER\_IF\_MODIFIED\_SINCE,

HTTPHEADER\_RANGE, HTTPHEADER\_REFERRER, HTTPHEADER\_REFERER, HTTPHEADER\_USER\_AGENT,

HTTPHEADER\_END\_OF\_ITEMS }

Enumerator for the IDs of httpHeader so we can know what the result was.

## **Functions**

int scanFor\_httpHeader (char \*str, unsigned int strLength)
 Scan a string for one of the words of the httpHeader word set.

## 6.52.1 Detailed Description

A tool that scans for a string in a very fast and robust way.

**Author** 

Ammar Qammaz (AmmarkoV)

## 6.52.2 Enumeration Type Documentation

6.52.2.1 anonymous enum

Enumerator for the IDs of httpHeader so we can know what the result was.

### **Enumerator**

HTTPHEADER\_EMPTY
HTTPHEADER\_AUTHORIZATION
HTTPHEADER\_ACCEPT\_ENCODING

HTTPHEADER\_COOKIE

HTTPHEADER\_CONNECTION

HTTPHEADER\_HOST

HTTPHEADER\_IF\_NONE\_MATCH

HTTPHEADER\_IF\_MODIFIED\_SINCE

HTTPHEADER\_RANGE

HTTPHEADER\_REFERRER

HTTPHEADER\_REFERER

HTTPHEADER\_USER\_AGENT

HTTPHEADER\_END\_OF\_ITEMS

## 6.52.3 Function Documentation

6.52.3.1 int scanFor\_httpHeader ( char \* str, unsigned int strLength )

Scan a string for one of the words of the httpHeader word set.

### **Parameters**

Input	String, to be scanned
Length	of Input String

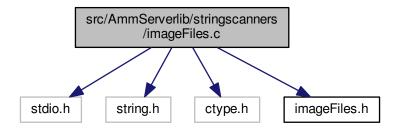
### Return values

See	above enumerator

## 6.53 src/AmmServerlib/stringscanners/imageFiles.c File Reference

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>
#include "imageFiles.h"
```

Include dependency graph for imageFiles.c:



## **Functions**

int scanFor\_imageFiles (char \*str, unsigned int strLength)
 Scan a string for one of the words of the imageFiles word set.

## 6.53.1 Function Documentation

6.53.1.1 int scanFor\_imageFiles ( char \* str, unsigned int strLength )

Scan a string for one of the words of the imageFiles word set.

### **Parameters**

Input	String, to be scanned
Length	of Input String

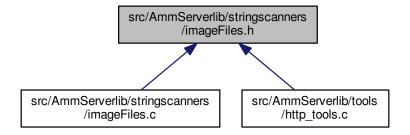
### **Return values**

See	above enumerator

## 6.54 src/AmmServerlib/stringscanners/imageFiles.h File Reference

A tool that scans for a string in a very fast and robust way.

This graph shows which files directly or indirectly include this file:



## **Enumerations**

• enum {

IMAGEFILES\_EMPTY =0, IMAGEFILES\_GIF, IMAGEFILES\_PNG, IMAGEFILES\_JPG, IMAGEFILES\_JPEG, IMAGEFILES\_WEBP, IMAGEFILES\_BMP, IMAGEFILES\_TIFF, IMAGEFILES\_DIB, IMAGEFILES\_RLE, IMAGEFILES\_J2C, IMAGEFILES\_ICO, IMAGEFILES\_PPM, IMAGEFILES\_PNM, IMAGEFILES\_RAW, IMAGEFILES\_SVG, IMAGEFILES\_END\_OF\_ITEMS }

Enumerator for the IDs of imageFiles so we can know what the result was.

### **Functions**

int scanFor\_imageFiles (char \*str, unsigned int strLength)
 Scan a string for one of the words of the imageFiles word set.

### 6.54.1 Detailed Description

A tool that scans for a string in a very fast and robust way.

### **Author**

Ammar Qammaz (AmmarkoV)

## 6.54.2 Enumeration Type Documentation

### 6.54.2.1 anonymous enum

Enumerator for the IDs of imageFiles so we can know what the result was.

### Enumerator

IMAGEFILES\_EMPTY IMAGEFILES\_GIF IMAGEFILES\_PNG IMAGEFILES\_JPG IMAGEFILES\_JPEG IMAGEFILES\_WEBP IMAGEFILES\_BMP IMAGEFILES\_TIFF IMAGEFILES\_DIB IMAGEFILES\_RLE IMAGEFILES\_J2C IMAGEFILES\_ICO IMAGEFILES\_PPM IMAGEFILES\_PNM **IMAGEFILES RAW** IMAGEFILES\_SVG IMAGEFILES\_END\_OF\_ITEMS

### 6.54.3 Function Documentation

6.54.3.1 int scanFor\_imageFiles ( char \* str, unsigned int strLength )

Scan a string for one of the words of the imageFiles word set.

### **Parameters**

Input	String , to be scanned
Length	of Input String

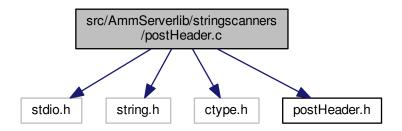
### Return values

See	above enumerator

## 6.55 src/AmmServerlib/stringscanners/postHeader.c File Reference

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>
#include "postHeader.h"
```

Include dependency graph for postHeader.c:



### **Functions**

• int scanFor\_postHeader (char \*str, unsigned int strLength)

Scan a string for one of the words of the postHeader word set.

### 6.55.1 Function Documentation

6.55.1.1 int scanFor\_postHeader ( char \* str, unsigned int strLength )

Scan a string for one of the words of the postHeader word set.

### **Parameters**

Input	String, to be scanned
Length	of Input String

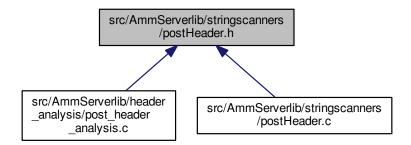
### Return values

See	above enumerator
-----	------------------

# 6.56 src/AmmServerlib/stringscanners/postHeader.h File Reference

A tool that scans for a string in a very fast and robust way.

This graph shows which files directly or indirectly include this file:



### **Enumerations**

enum {
 POSTHEADER\_EMPTY =0, POSTHEADER\_CONTENT\_TYPE, POSTHEADER\_CONTENT\_DISPOSITION, POSTHEADER\_CONTENT\_LENGTH,
 POSTHEADER\_END\_OF\_ITEMS }

Enumerator for the IDs of postHeader so we can know what the result was.

## **Functions**

int scanFor\_postHeader (char \*str, unsigned int strLength)
 Scan a string for one of the words of the postHeader word set.

## 6.56.1 Detailed Description

A tool that scans for a string in a very fast and robust way.

Author

Ammar Qammaz (AmmarkoV)

## 6.56.2 Enumeration Type Documentation

6.56.2.1 anonymous enum

Enumerator for the IDs of postHeader so we can know what the result was.

### Enumerator

POSTHEADER\_EMPTY
POSTHEADER\_CONTENT\_TYPE
POSTHEADER\_CONTENT\_DISPOSITION
POSTHEADER\_CONTENT\_LENGTH
POSTHEADER\_END\_OF\_ITEMS

## 6.56.3 Function Documentation

6.56.3.1 int scanFor\_postHeader ( char \* str, unsigned int strLength )

Scan a string for one of the words of the postHeader word set.

### **Parameters**

Input	String , to be scanned
Length	of Input String

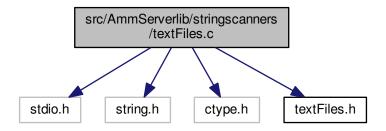
### Return values

See	above enumerator

## 6.57 src/AmmServerlib/stringscanners/textFiles.c File Reference

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>
#include "textFiles.h"
```

Include dependency graph for textFiles.c:



### **Functions**

• int scanFor\_textFiles (char \*str, unsigned int strLength)

Scan a string for one of the words of the textFiles word set.

## 6.57.1 Function Documentation

6.57.1.1 int scanFor\_textFiles ( char \* str, unsigned int strLength )

Scan a string for one of the words of the textFiles word set.

## **Parameters**

Input	String, to be scanned
Length	of Input String

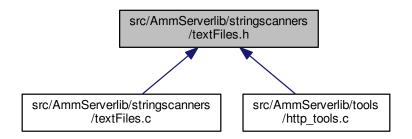
### Return values

See	above enumerator

## 6.58 src/AmmServerlib/stringscanners/textFiles.h File Reference

A tool that scans for a string in a very fast and robust way.

This graph shows which files directly or indirectly include this file:



### **Enumerations**

enum {
 TEXTFILES\_EMPTY =0, TEXTFILES\_HTML, TEXTFILES\_LTML, TEXTFILES\_CSS,
 TEXTFILES\_TXT, TEXTFILES\_DOC, TEXTFILES\_RTF, TEXTFILES\_ODF,
 TEXTFILES\_ODT, TEXTFILES\_END\_OF\_ITEMS }

Enumerator for the IDs of textFiles so we can know what the result was.

## **Functions**

int scanFor\_textFiles (char \*str, unsigned int strLength)
 Scan a string for one of the words of the textFiles word set.

## 6.58.1 Detailed Description

A tool that scans for a string in a very fast and robust way.

Author

Ammar Qammaz (AmmarkoV)

## 6.58.2 Enumeration Type Documentation

6.58.2.1 anonymous enum

Enumerator for the IDs of textFiles so we can know what the result was.

## Enumerator

TEXTFILES\_EMPTY
TEXTFILES\_HTML
TEXTFILES\_HTM
TEXTFILES\_CSS

TEXTFILES\_TXT

TEXTFILES\_DOC

TEXTFILES\_RTF

TEXTFILES\_ODF

TEXTFILES\_ODT

TEXTFILES\_END\_OF\_ITEMS

### 6.58.3 Function Documentation

6.58.3.1 int scanFor\_textFiles ( char \* str, unsigned int strLength )

Scan a string for one of the words of the textFiles word set.

### **Parameters**

Input	String , to be scanned
Length	of Input String

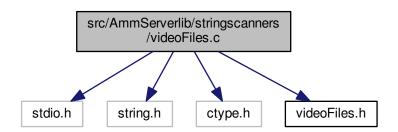
### Return values

See	above enumerator
-----	------------------

## 6.59 src/AmmServerlib/stringscanners/videoFiles.c File Reference

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>
#include "videoFiles.h"
```

Include dependency graph for videoFiles.c:



### **Functions**

• int scanFor\_videoFiles (char \*str, unsigned int strLength)

Scan a string for one of the words of the videoFiles word set.

## 6.59.1 Function Documentation

6.59.1.1 int scanFor\_videoFiles ( char \* str, unsigned int strLength )

Scan a string for one of the words of the videoFiles word set.

### **Parameters**

Input	String , to be scanned
Length	of Input String

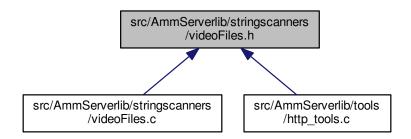
### Return values

See	above enumerator

# 6.60 src/AmmServerlib/stringscanners/videoFiles.h File Reference

A tool that scans for a string in a very fast and robust way.

This graph shows which files directly or indirectly include this file:



## **Enumerations**

enum {
 VIDEOFILES\_EMPTY =0, VIDEOFILES\_AVI, VIDEOFILES\_MPEG4, VIDEOFILES\_MPEG,
 VIDEOFILES\_MP4, VIDEOFILES\_WEBM, VIDEOFILES\_MKV, VIDEOFILES\_3GP,
 VIDEOFILES\_H263, VIDEOFILES\_H264, VIDEOFILES\_FLV, VIDEOFILES\_END\_OF\_ITEMS }

Enumerator for the IDs of videoFiles so we can know what the result was.

## **Functions**

int scanFor\_videoFiles (char \*str, unsigned int strLength)

Scan a string for one of the words of the videoFiles word set.

## 6.60.1 Detailed Description

A tool that scans for a string in a very fast and robust way.

### **Author**

Ammar Qammaz (AmmarkoV)

## 6.60.2 Enumeration Type Documentation

## 6.60.2.1 anonymous enum

Enumerator for the IDs of videoFiles so we can know what the result was.

### **Enumerator**

VIDEOFILES\_EMPTY

VIDEOFILES\_AVI

VIDEOFILES\_MPEG4

VIDEOFILES\_MPEG

VIDEOFILES\_MP4

VIDEOFILES\_WEBM

VIDEOFILES MKV

VIDEOFILES\_3GP

VIDEOFILES\_H263

VIDEOFILES\_H264

VIDEOFILES\_FLV

VIDEOFILES\_END\_OF\_ITEMS

### 6.60.3 Function Documentation

```
6.60.3.1 int scanFor_videoFiles ( char * str, unsigned int strLength )
```

Scan a string for one of the words of the videoFiles word set.

## Parameters

Input	String, to be scanned
Length	of Input String

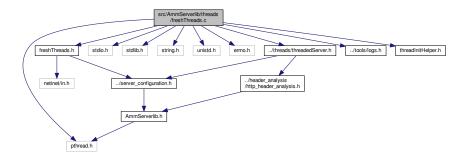
### Return values

See	above enumerator

## 6.61 src/AmmServerlib/threads/freshThreads.c File Reference

```
#include "freshThreads.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <pthread.h>
#include <errno.h>
#include "../threads/threadedServer.h"
#include "../tools/logs.h"
#include "threadInitHelper.h"
```

Include dependency graph for freshThreads.c:



### **Macros**

- #define WEIRD\_THING\_THAT\_WORKS 1
- #define MAX\_TRIES\_TO\_FIND\_A\_THREAD\_ID 5

### **Functions**

- unsigned int FindAProperThreadID (struct AmmServer\_Instance \*instance, int \*success)
- int SpawnThreadToServeNewClient (struct AmmServer\_Instance \*instance, int clientsock, struct sockaddr\_in client, unsigned int clientlen, char \*webserver root, char \*templates root)

Create a new Thread that will serve the incoming client socket connection.

### 6.61.1 Macro Definition Documentation

- 6.61.1.1 #define MAX\_TRIES\_TO\_FIND\_A\_THREAD\_ID 5
- 6.61.1.2 #define WEIRD\_THING\_THAT\_WORKS 1

### 6.61.2 Function Documentation

6.61.2.1 unsigned int FindAProperThreadID ( struct AmmServer\_Instance \* instance, int \* success )

Here is the call graph for this function:



6.61.2.2 int SpawnThreadToServeNewClient ( struct AmmServer\_Instance \* instance, int clientsock, struct sockaddr\_in client, unsigned int clientlen, char \* webserver\_root, char \* templates\_root )

Create a new Thread that will serve the incoming client socket connection.

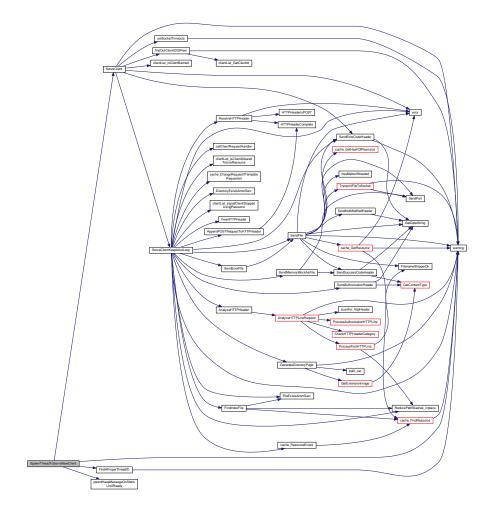
### **Parameters**

An	AmmarServer Instance
Client	socket to be read
Client	socket to be read ( sockaddr_in )
Length	of client
Filename	of root directory for this connection ( public_html )
Filename	of template directory for this connection ( for 404.html etc )

## Return values

1=Success,0=Fail	

Here is the call graph for this function:

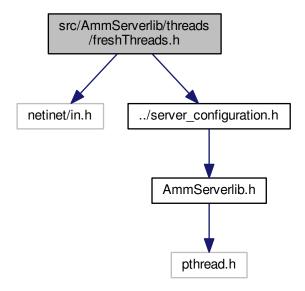


# 6.62 src/AmmServerlib/threads/freshThreads.h File Reference

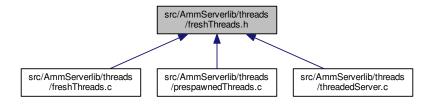
Creating new threads to serve clients, we only have one call that generates a thread that serves a client connection.

```
#include <netinet/in.h>
#include "../server_configuration.h"
```

Include dependency graph for freshThreads.h:



This graph shows which files directly or indirectly include this file:



## **Data Structures**

struct PassToHTTPThread

A structure that holds information to be passed from the main thread to the new (fresh) thread.

## **Functions**

• int SpawnThreadToServeNewClient (struct AmmServer\_Instance \*instance, int clientsock, struct sockaddr\_in client, unsigned int clientlen, char \*webserver\_root, char \*templates\_root)

Create a new Thread that will serve the incoming client socket connection.

## 6.62.1 Detailed Description

Creating new threads to serve clients, we only have one call that generates a thread that serves a client connection.

Author

Ammar Qammaz (AmmarkoV)

## 6.62.2 Function Documentation

6.62.2.1 int SpawnThreadToServeNewClient ( struct AmmServer\_Instance \* instance, int clientsock, struct sockaddr\_in client, unsigned int clientlen, char \* webserver\_root, char \* templates\_root )

Create a new Thread that will serve the incoming client socket connection.

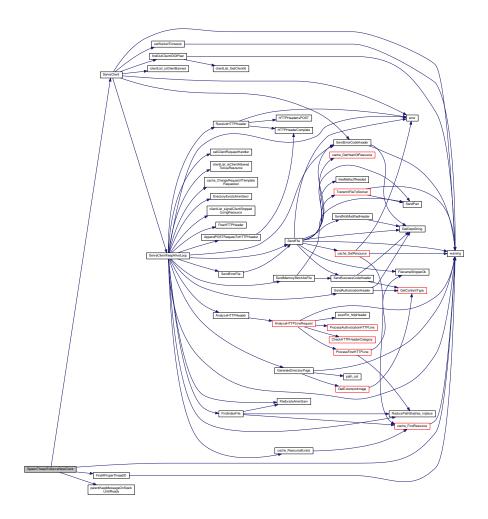
### **Parameters**

An	AmmarServer Instance
Client	socket to be read
Client	socket to be read ( sockaddr_in )
Length	of client
Filename	of root directory for this connection ( public_html )
Filename	of template directory for this connection ( for 404.html etc )

### Return values

1=Success,0=Fail	
------------------	--

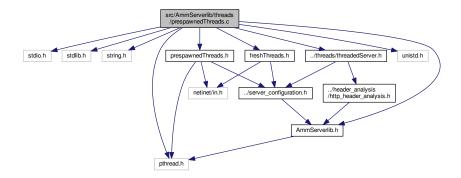
Here is the call graph for this function:



## 6.63 src/AmmServerlib/threads/prespawnedThreads.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "prespawnedThreads.h"
#include "freshThreads.h"
#include <pthread.h>
#include <unistd.h>
#include "../threads/threadedServer.h"
#include "../AmmServerlib.h"
```

Include dependency graph for prespawnedThreads.c:



## **Data Structures**

• struct PassToPreSpawnedThread

### **Functions**

- void \* PreSpawnedThread (void \*ptr)
- void PreSpawnThreads (struct AmmServer\_Instance \*instance)

Create an initial pool of PreSpawned Threads , before handling any connections to be ready when a connection arrives.

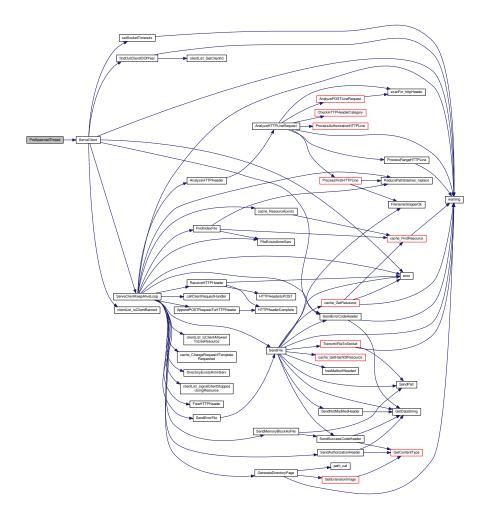
• int UsePreSpawnedThreadToServeNewClient (struct AmmServer\_Instance \*instance, int clientsock, struct sockaddr\_in client, unsigned int clientlen, char \*webserver\_root, char \*templates\_root)

Use a PreSpawned Thread that will serve the incoming client socket connection.

## 6.63.1 Function Documentation

## 6.63.1.1 void\* PreSpawnedThread (void \* ptr)

Here is the call graph for this function:



## 6.63.1.2 void PreSpawnThreads ( struct AmmServer\_Instance \* instance )

Create an initial pool of PreSpawned Threads , before handling any connections to be ready when a connection arrives.

## **Parameters**

An AmmarServer Instance	
-------------------------	--

## Return values

1=Success,0=Fail	

6.63.1.3 int UsePreSpawnedThreadToServeNewClient ( struct AmmServer\_Instance \* instance, int clientsock, struct sockaddr\_in client, unsigned int clientlen, char \* webserver\_root, char \* templates\_root )

Use a PreSpawned Thread that will serve the incoming client socket connection.

### **Parameters**

An	AmmarServer Instance
Client	socket to be read
Client	socket to be read ( sockaddr_in )
Length	of client
Filename	of root directory for this connection ( public_html )
Filename	of template directory for this connection ( for 404.html etc )

## Return values

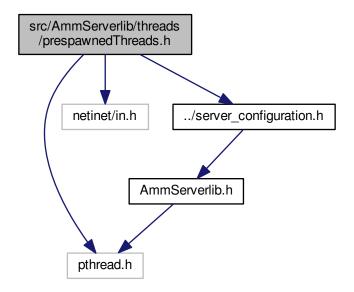
1=Success,0=Fail	

# 6.64 src/AmmServerlib/threads/prespawnedThreads.h File Reference

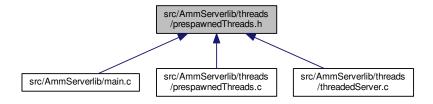
Using already created threads to serve clients, we have a pool of threads that can be used to serve connections.

```
#include <pthread.h>
#include <netinet/in.h>
#include "../server_configuration.h"
```

Include dependency graph for prespawnedThreads.h:



This graph shows which files directly or indirectly include this file:



### **Data Structures**

struct PreSpawnedThread

A structure that holds information to be passed from the main thread to the new (prespawned) thread.

### **Functions**

- void PreSpawnThreads (struct AmmServer\_Instance \*instance)
   Create an initial pool of PreSpawned Threads, before handling any connections to be ready when a connection arrives.
- int UsePreSpawnedThreadToServeNewClient (struct AmmServer\_Instance \*instance, int clientsock, struct sockaddr\_in client, unsigned int clientlen, char \*webserver\_root, char \*templates\_root)

Use a PreSpawned Thread that will serve the incoming client socket connection.

## 6.64.1 Detailed Description

Using already created threads to serve clients, we have a pool of threads that can be used to serve connections.

**Author** 

Ammar Qammaz (AmmarkoV)

Bug Prespawned threads have race conditions?

## 6.64.2 Function Documentation

6.64.2.1 void PreSpawnThreads ( struct AmmServer\_Instance \* instance )

Create an initial pool of PreSpawned Threads , before handling any connections to be ready when a connection arrives.

**Parameters** 

An	AmmarServer Instance

Return values

```
1=Success,0=Fail
```

6.64.2.2 int UsePreSpawnedThreadToServeNewClient ( struct AmmServer\_Instance \* instance, int clientsock, struct sockaddr\_in client, unsigned int clientlen, char \* webserver\_root, char \* templates\_root )

Use a PreSpawned Thread that will serve the incoming client socket connection.

#### **Parameters**

An	AmmarServer Instance
Client	socket to be read
Client	socket to be read ( sockaddr_in )
Length	of client
Filename	of root directory for this connection ( public_html )
Filename	of template directory for this connection ( for 404.html etc )

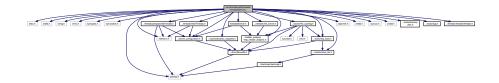
#### Return values

```
1=Success,0=Fail
```

## 6.65 src/AmmServerlib/threads/threadedServer.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <errno.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <sys/uio.h>
#include <unistd.h>
#include <pthread.h>
#include "threadedServer.h"
#include "../tools/directory_lists.h"
#include "../network/file_server.h"
#include "../header_analysis/http_header_analysis.h"
#include "../tools/http_tools.h"
#include "../tools/logs.h"
#include "../cache/file caching.h"
#include "../server_configuration.h"
#include "../threads/freshThreads.h"
#include "../threads/prespawnedThreads.h"
#include "../threads/threadInitHelper.h"
#include "../cache/client_list.h"
#include "../cache/dynamic_requests.h"
```

Include dependency graph for threadedServer.c:



## **Functions**

• int HTTPServerIsRunning (struct AmmServer\_Instance \*instance)

Ask if the HTTP server is running.

- int ServeClientKeepAliveLoop (struct AmmServer\_Instance \*instance, struct HTTPTransaction \*transaction)
- void \* ServeClient (void \*ptr)

Main Call to Serve a client, this will in turn pick a prespawned thread or create a new one.

- void \* MainHTTPServerThread (void \*ptr)
- int StartHTTPServer (struct AmmServer\_Instance \*instance, char \*ip, unsigned int port, char \*root\_path, char \*templates\_path)

Start HTTP server.

• int StopHTTPServer (struct AmmServer\_Instance \*instance)

Stop a running HTTP server , unbind ports , deallocate structures etc.

## 6.65.1 Function Documentation

6.65.1.1 int HTTPServerlsRunning ( struct AmmServer Instance \* instance )

Ask if the HTTP server is running.

**Parameters** 

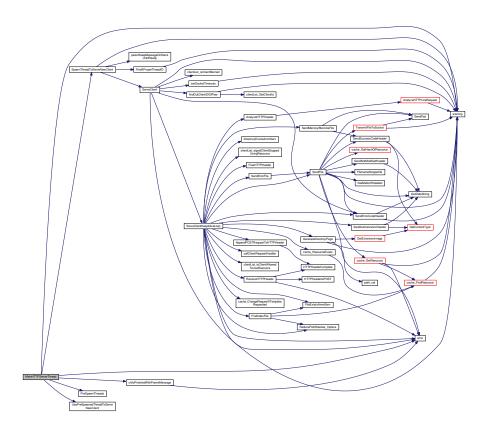
	An	AmmarServer Instance
--	----	----------------------

**Return values** 

1=Success,0=Failure	

6.65.1.2 void\* MainHTTPServerThread ( void \* ptr )

Here is the call graph for this function:



6.65.1.3 void\* ServeClient (void \* ptr)

Main Call to Serve a client, this will in turn pick a prespawned thread or create a new one.

## **Parameters**

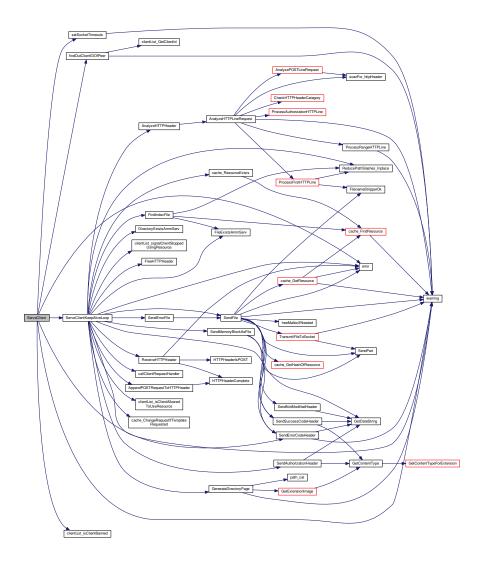
PassToHTTP-	with information to pass to the new thread ( prespawned or not )
Thread	

## Return values

This	function returns 0

START OF CLIENT IS NOT ON IP-BANNED-LIST!

Here is the call graph for this function:

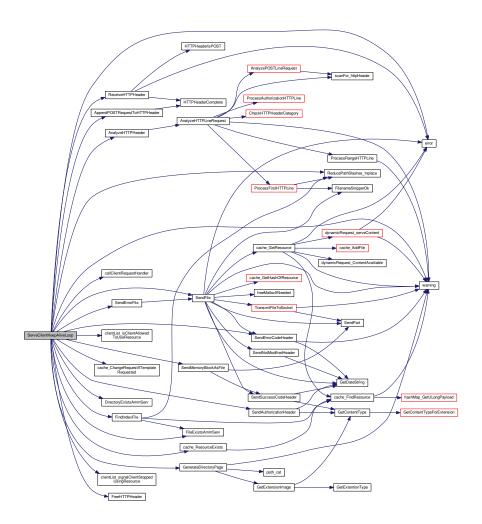


6.65.1.4 int ServeClientKeepAliveLoop ( struct AmmServer\_Instance \* instance, struct HTTPTransaction \* transaction \* in [inline]

PART 1 : Sense what we want to serve , and set the flags resource\_is\_a\_directory , resource\_is\_a\_file , generate\_directory\_list accordingly..!

PART 2 : The flags resource\_is\_a\_directory , resource\_is\_a\_file , generate\_directory\_list have been set to the correct (:P) value so all we have to do now is serve the correct repsonse..!

Here is the call graph for this function:



6.65.1.5 int StartHTTPServer ( struct AmmServer\_Instance \* instance, char \* ip, unsigned int port, char \* root\_path, char \* templates\_path )

## Start HTTP server.

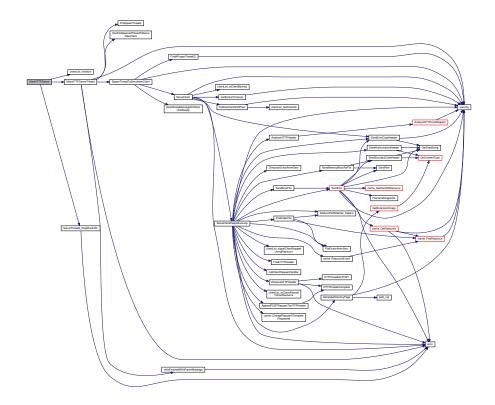
### **Parameters**

An	AmmarServer Instance
String	with the binding IP for the new server
Port	for binding the new server , ports under 1000 require super user privileges
Filename	to root path for this webserver ( public_html )
Filename	to root path for templates ( 404.html etc )

### Return values

1=Success,0=Failure
---------------------

Here is the call graph for this function:



## 6.65.1.6 int StopHTTPServer ( struct AmmServer\_Instance \* instance )

Stop a running HTTP server , unbind ports , deallocate structures etc.

### **Parameters**

An	AmmarServer Instance

Bug Stop web server should be improved , to make sure it unbinds the closing socket

## Return values

1=Success,0=Failure

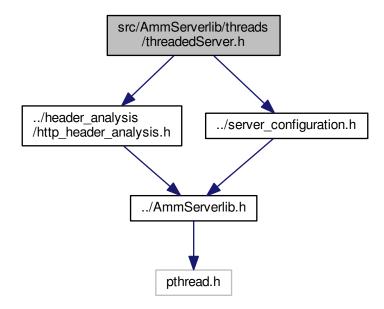
Here is the call graph for this function:



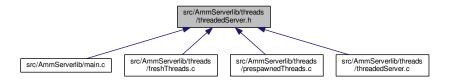
## 6.66 src/AmmServerlib/threads/threadedServer.h File Reference

Creating new threads to serve clients, we only have one call that generates a thread that serves a client connection.

```
#include "../header_analysis/http_header_analysis.h"
#include "../server_configuration.h"
Include dependency graph for threadedServer.h:
```



This graph shows which files directly or indirectly include this file:



### **Functions**

- void \* ServeClient (void \*ptr)
  - Main Call to Serve a client, this will in turn pick a prespawned thread or create a new one.
- int StartHTTPServer (struct AmmServer\_Instance \*instance, char \*ip, unsigned int port, char \*root\_path, char \*templates\_path)

Start HTTP server.

- int StopHTTPServer (struct AmmServer Instance \*instance)
  - Stop a running HTTP server , unbind ports , deallocate structures etc.
- int HTTPServerIsRunning (struct AmmServer\_Instance \*instance)

Ask if the HTTP server is running.

## 6.66.1 Detailed Description

Creating new threads to serve clients, we only have one call that generates a thread that serves a client connection.

**Author** 

Ammar Qammaz (AmmarkoV)

### 6.66.2 Function Documentation

6.66.2.1 int HTTPServerlsRunning ( struct AmmServer\_Instance \* instance )

Ask if the HTTP server is running.

**Parameters** 

An	AmmarServer Instance
----	----------------------

Return values

```
1=Success,0=Failure
```

6.66.2.2 void\* ServeClient (void \* ptr)

Main Call to Serve a client , this will in turn pick a prespawned thread or create a new one.

**Parameters** 

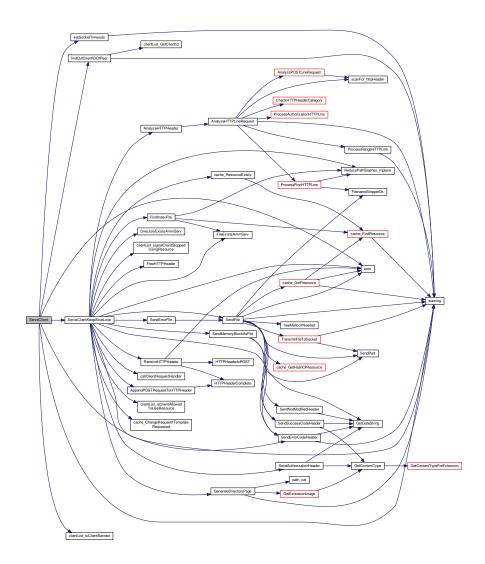
PassToHTTP-	with information to pass to the new thread ( prespawned or not )
Thread	

Return values

This	function returns 0

START OF CLIENT IS NOT ON IP-BANNED-LIST!

Here is the call graph for this function:



6.66.2.3 int StartHTTPServer ( struct AmmServer\_Instance \* instance, char \* ip, unsigned int port, char \* root\_path, char \* templates\_path )

## Start HTTP server.

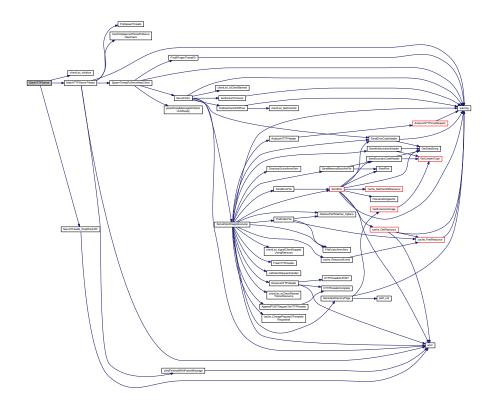
### **Parameters**

An	AmmarServer Instance
String	with the binding IP for the new server
Port for binding the new server, ports under 1000 require super user privileges	
Filename	to root path for this webserver ( public_html )
Filename	to root path for templates ( 404.html etc )

### Return values

1=Success,0=Failure	
---------------------	--

Here is the call graph for this function:



## 6.66.2.4 int StopHTTPServer ( struct AmmServer\_Instance \* instance )

Stop a running HTTP server , unbind ports , deallocate structures etc.

### **Parameters**

An	AmmarServer Instance

Bug Stop web server should be improved , to make sure it unbinds the closing socket

## Return values

1=Success,0=Failure

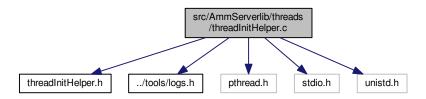
Here is the call graph for this function:



## 6.67 src/AmmServerlib/threads/threadInitHelper.c File Reference

```
#include "threadInitHelper.h"
#include "../tools/logs.h"
#include <pthread.h>
#include <stdio.h>
#include <unistd.h>
```

Include dependency graph for threadInitHelper.c:



### **Macros**

• #define SLEEP\_FOR\_N\_NANOSECONDS\_WAITING\_STACK\_MESSAGE 10

### **Functions**

- int parentKeepMessageOnStackUntilReadyOrTimeout (volatile int \*childSwitch, unsigned int maxWaitTime)

  A call to help waiting for a message to be consumed by a child thread, or until a timeout, this should be called by the parent thread.
- int parentKeepMessageOnStackUntilReady (volatile int \*childSwitch)

A call to help waiting for a message to be consumed by a child thread, this should be called by the parent thread.

void childFinishedWithParentMessage (volatile int \*childSwitch)

A call to help waiting for a message to be consumed by a child thread, this should be called by the child thread when it finished copying the message.

### 6.67.1 Macro Definition Documentation

6.67.1.1 #define SLEEP\_FOR\_N\_NANOSECONDS\_WAITING\_STACK\_MESSAGE 10

## 6.67.2 Function Documentation

 $6.67.2.1 \quad \text{void childFinishedWithParentMessage (} \ \ \text{volatile int} * \textit{childSwitch} \ \ )$ 

A call to help waiting for a message to be consumed by a child thread , this should be called by the child thread when it finished copying the message.

### **Parameters**

Pointer	to the switch signaling that the child has read the message , so that the parent will keep it in	
	his stack	

### Return values

1=Success,0=Fail

Here is the call graph for this function:

childFinishedWithParentMessage error

## 6.67.2.2 int parentKeepMessageOnStackUntilReady ( volatile int \* childSwitch )

A call to help waiting for a message to be consumed by a child thread, this should be called by the parent thread.

### **Parameters**

Pointer	to the switch signaling that the child has read the message, so that the parent will keep it in
	his stack

## Return values

1=Success,0=Fail	

## 6.67.2.3 int parentKeepMessageOnStackUntilReadyOrTimeout ( volatile int \* childSwitch, unsigned int maxWaitTime )

A call to help waiting for a message to be consumed by a child thread , or until a timeout , this should be called by the parent thread.

## Parameters

Pointer	to the switch signaling that the child has read the message, so that the parent will keep it in	
	his stack	
Maximum	time to wait before terminating the wait to ensure no dead-locks	

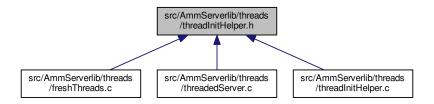
## Return values

1=Success,0=Fail	

# 6.68 src/AmmServerlib/threads/threadInitHelper.h File Reference

Helper Functions to help with passing messages around ..

This graph shows which files directly or indirectly include this file:



### **Functions**

• int parentKeepMessageOnStackUntilReadyOrTimeout (volatile int \*childSwitch, unsigned int maxWaitTime)

A call to help waiting for a message to be consumed by a child thread, or until a timeout, this should be called by the parent thread.

• int parentKeepMessageOnStackUntilReady (volatile int \*childSwitch)

A call to help waiting for a message to be consumed by a child thread, this should be called by the parent thread.

void childFinishedWithParentMessage (volatile int \*childSwitch)

A call to help waiting for a message to be consumed by a child thread, this should be called by the child thread when it finished copying the message.

## 6.68.1 Detailed Description

Helper Functions to help with passing messages around ..

Author

Ammar Qammaz (AmmarkoV)

### 6.68.2 Function Documentation

6.68.2.1 void childFinishedWithParentMessage ( volatile int \* childSwitch )

A call to help waiting for a message to be consumed by a child thread , this should be called by the child thread when it finished copying the message.

## **Parameters**

Pointer	to the switch signaling that the child has read the message, so that the parent will keep it in
	his stack

### **Return values**

1=Success,0=Fail
------------------

Here is the call graph for this function:



## 6.68.2.2 int parentKeepMessageOnStackUntilReady ( volatile int \* childSwitch )

A call to help waiting for a message to be consumed by a child thread, this should be called by the parent thread.

### **Parameters**

Pointer	to the switch signaling that the child has read the message, so that the parent will keep it in
	his stack

### Return values

```
1=Success,0=Fail
```

 $6.68.2.3 \quad \text{int parentKeepMessageOnStackUntilReadyOrTimeout ( \ volatile \ int } * \textit{childSwitch, } \ unsigned \ int \ \textit{maxWaitTime } )$ 

A call to help waiting for a message to be consumed by a child thread, or until a timeout, this should be called by the parent thread.

### **Parameters**

Pointer	to the switch signaling that the child has read the message , so that the parent will keep it in his stack
Maximum	time to wait before terminating the wait to ensure no dead-locks

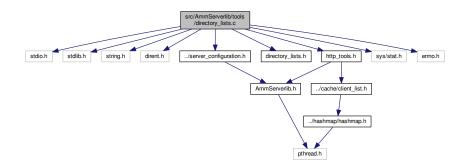
## Return values

```
1=Success,0=Fail
```

## 6.69 src/AmmServerlib/tools/directory\_lists.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <dirent.h>
#include "../server_configuration.h"
#include "directory_lists.h"
#include "http_tools.h"
#include <sys/stat.h>
#include <errno.h>
```

Include dependency graph for directory\_lists.c:



#### **Macros**

- #define starting "<html>\ <head>\ <meta http-equiv=\"Content-Type\" content=\"text/html; charset=UTF-8\" />\ <title>AmmarServer Directory listing</title>\ </head>\ <body>\ <h1>AmmarServer Directory Listing</h1>\ <a name=\"top\"></a><hr>\ <!-- TODO: Add some clientside javascript search and sorting capabilities! :P -->\ \ \ \ <a href=\"#todo2\">Filename</a><a href=\"#todo3\">Byte Size</a><a href=\"#todo4\">Modification Date</a>\ </rr>
- #define tag pre image "<img src=\"/"</li>
- #define tag\_after\_image "\">"

#### **Functions**

- char \* path\_cat (const char \*str1, char \*str2)
- char \* GenerateDirectoryPage (char \*system\_path, char \*client\_path, unsigned long \*memoryUsed)

  Return a memory buffer containing the contents o a directory listing.

### 6.69.1 Macro Definition Documentation

- 6.69.1.2 #define tag\_after\_image "\">"
- 6.69.1.3 #define tag\_pre\_image "<img src=\"/"

#### 6.69.2 Function Documentation

6.69.2.1 char\* GenerateDirectoryPage ( char \* system\_path, char \* client\_path, unsigned long \* memoryUsed )

Return a memory buffer containing the contents o a directory listing.

**Parameters** 

System	path to list
Client	path ( relative to root directory of client etc )
Input	size of memory tou allocate and Output size of memory used

#### Return values

Pointer	to memory that contains directory listing ,0=Failure

Bug GenerateDirectoryPage does not handle memory correctly , code is in very bad shape

Here is the call graph for this function:

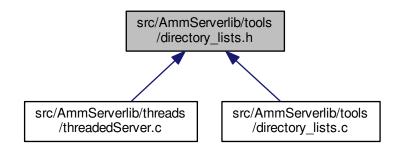


6.69.2.2 char\* path\_cat ( const char \* str1, char \* str2 )

# 6.70 src/AmmServerlib/tools/directory\_lists.h File Reference

Basic file server functionality of AmmarServer.

This graph shows which files directly or indirectly include this file:



#### **Functions**

char \* GenerateDirectoryPage (char \*system\_path, char \*client\_path, unsigned long \*memoryUsed)
 Return a memory buffer containing the contents o a directory listing.

# 6.70.1 Detailed Description

Basic file server functionality of AmmarServer.

Author

Ammar Qammaz (AmmarkoV)

# 6.70.2 Function Documentation

6.70.2.1 char\* GenerateDirectoryPage ( char\* system\_path, char\* client\_path, unsigned long\* memoryUsed )

Return a memory buffer containing the contents o a directory listing.

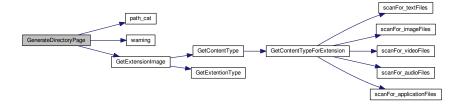
#### **Parameters**

ſ	System	path to list
ĺ	Client	path ( relative to root directory of client etc )
ĺ	Input	size of memory tou allocate and Output size of memory used

#### **Return values**

Pointer	to memory that contains directory listing ,0=Failure

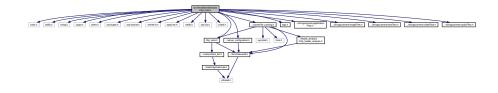
Bug GenerateDirectoryPage does not handle memory correctly , code is in very bad shape



# 6.71 src/AmmServerlib/tools/http\_tools.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <dirent.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <sys/uio.h>
#include <unistd.h>
#include "http tools.h"
#include "logs.h"
#include "../server_configuration.h"
#include "../cache/file_caching.h"
#include "../stringscanners/applicationFiles.h"
#include "../stringscanners/imageFiles.h"
#include "../stringscanners/textFiles.h"
#include "../stringscanners/videoFiles.h"
#include "../stringscanners/audioFiles.h"
```

Include dependency graph for http tools.c:



#### **Functions**

unsigned int ServerThreads\_DropRootUID ()

Drop Root UID, if we have one (and according to server\_configuration.h)

char FileExistsAmmServ (char \*filename)

Check if file Exists.

char DirectoryExistsAmmServ (char \*dirpath)

Check if directory Exists.

- int GetContentTypeForExtension (char \*theextension, char \*content\_type, unsigned int contentTypeLength)
- int GetExtentionType (char \*theextension)

Convert an Extension Type to a contentTypeEnumerator.

- void convertToUpperCase (char \*sPtr)
- int GetContentType (char \*filename, char \*contentType, unsigned int contentTypeLength)

Convert a filename to a contentType.

• int GetExtensionImage (char \*filename, char \*theimagepath, unsigned int theimagepath length)

Return template image for specific content type ( for directory listings etc )

int ReducePathSlashes\_Inplace (char \*filename)

Filenames may contain //// with an arbitrary number of slashes , we convert them to a single slash ,.

- int StripGETRequestQueryAndFragment (char \*filename, char \*query, unsigned int max\_query\_length)
- int StripVariableFromGETorPOSTString (char \*input, char \*var\_id, char \*var\_val, unsigned int var\_val\_length)
- int StripHTMLCharacters\_Inplace (char \*filename, int enable\_security)

HTML characters should be converted to plain c byte chars after we get them, this poses some security threats since this might allow "weird" bytes to get set that in conjunction with an overflow somewhere else might trick the server into executing,.

• int FilenameStripperOk (char \*filename)

Strip filename and security check it.

- int strToUpcase (char \*strTarget, char \*strSource, unsigned int strLength)
- int stristr (char \*str1CAPS, unsigned int str1\_length, char \*str2CAPS, unsigned int str2\_length, unsigned int \*pos\_found)
- int stristr2Caps (char \*str1, unsigned int str1\_length, char \*str2CAPS, unsigned int str2\_length, unsigned int \*pos\_found)
- int trim\_last\_empty\_chars (char \*input, unsigned int input\_length)
- int seek non blank char (char \*input, char \*input end)
- int seek blank char (char \*input, char \*input end)
- unsigned int GetIntFromHTTPHeaderFieldPayload (char \*request, unsigned int request\_length)
- char \* GetNewStringFromHTTPHeaderFieldPayload (char \*request, unsigned int request length)
- int encodeToBase64 (char \*src, unsigned s\_len, char \*dst, unsigned d\_len)

Convert a string to base64, required for the authorization tokens.

- int CheckHTTPHeaderCategoryAllCaps (char \*lineCAPS, unsigned int line\_length, char \*potential\_strCAPS, unsigned int \*payload\_start)
- int CheckHTTPHeaderCategory (char \*line, unsigned int line\_length, char \*potential\_strCAPS, unsigned int \*payload\_start)
- int FindIndexFile (struct AmmServer\_Instance \*instance, char \*webserver\_root, char \*directory, char \*indexfile)
- char \* RequestHTTPWebPage (char \*hostname, unsigned int port, char \*filename, unsigned int max\_content)

A very basic http client for testing connections and maybe in the future make AmmarServers communicate with each other

int freeString (char \*\*str)

Free C string and set it to 0.

int setSocketTimeouts (int clientSock)

Enforce socket timeouts declared in server\_configuration.h and configuration files to socket.

clientID findOutClientIDOfPeer (struct AmmServer Instance \*instance, int clientSock)

Tool that resolve a client socket to its IP , then uses it to try to clientList\_GetClientId and returns the id number.

# 6.71.1 Function Documentation

6.71.1.1 int CheckHTTPHeaderCategory ( char \* line, unsigned int line\_length, char \* potential\_strCAPS, unsigned int \* payload\_start )



6.71.1.2 int CheckHTTPHeaderCategoryAllCaps ( char \* lineCAPS, unsigned int line\_length, char \* potential\_strCAPS, unsigned int \* payload\_start )

Here is the call graph for this function:



- 6.71.1.3 void convertToUpperCase ( char \* sPtr )
- 6.71.1.4 char DirectoryExistsAmmServ ( char \* dirpath )

Check if directory Exists.

#### **Parameters**

Path	to directory

#### Return values

1=Exists,0=Does	not Exist

6.71.1.5 int encodeToBase64 ( char \* src, unsigned s\_len, char \* dst, unsigned d\_len )

Convert a string to base64, required for the authorization tokens.

# **Parameters**

ſ	Input	string
	Input	string length
	Output	string
ĺ	Input	Maximum output length

# Return values

1=Success,0=Failure	

6.71.1.6 char FileExistsAmmServ ( char \* filename )

Check if file Exists.

### **Parameters**

Path	to file

#### Return values

1=Exists,0=Does	not Exist
-----------------	-----------

#### 6.71.1.7 int FilenameStripperOk ( char \* filename )

Strip filename and security check it.

#### **Parameters**

Pointer	to string pointer to be analyzed

#### Return values

```
1=Ok,0=Failed
```

# 6.71.1.8 int FindIndexFile ( struct AmmServer\_Instance \* instance, char \* webserver\_root, char \* directory, char \* indexfile )

Here is the call graph for this function:



# 6.71.1.9 clientID findOutClientIDOfPeer ( struct AmmServer\_Instance \* instance, int clientSock )

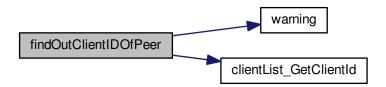
Tool that resolve a client socket to its IP , then uses it to try to clientList\_GetClientId and returns the id number.

#### **Parameters**

An	AmmarServer instance
client	socket

# Return values





6.71.1.10 int freeString ( char \*\* str )

Free C string and set it to 0.

# **Parameters**

Pointer	to string pointer to be freed
---------	-------------------------------

#### Return values

```
1=Success,0=Failure
```

6.71.1.11 int GetContentType ( char \* filename, char \* contentType, unsigned int contentTypeLength )

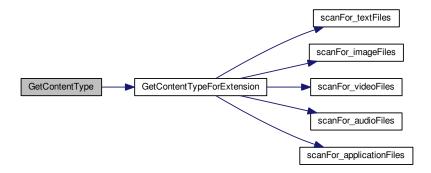
Convert a filename to a contentType.

#### **Parameters**

String	with the filename we want to examine
Output	String with the contentType
Output	contentType length

# Return values

contentTypeEnumerato	
----------------------	--



Here is the call graph for this function:

6.71.1.12 int GetContentTypeForExtension ( char \* theextension, char \* content\_type, unsigned int contentTypeLength )

scanFor\_textFiles

scanFor\_imageFiles

scanFor\_videoFiles

scanFor\_audioFiles

scanFor\_applicationFiles

6.71.1.13 int GetExtensionImage ( char \* filename, char \* theimagepath, unsigned int theimagepath\_length )

Return template image for specific content type ( for directory listings etc )

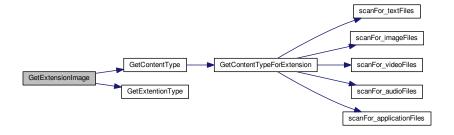
#### **Parameters**

Filename	of file
Path	to Image
Length	of path to Image

# Return values

1=Exists,0=Does	not Exist

Here is the call graph for this function:



6.71.1.14 int GetExtentionType ( char \* theextension )

Convert an Extension Type to a contentTypeEnumerator.

#### **Parameters**

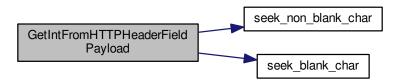
String	with the extension type

#### **Return values**

```
contentTypeEnumerator
```

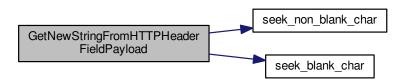
6.71.1.15 unsigned int GetIntFromHTTPHeaderFieldPayload ( char \* request, unsigned int request\_length )

Here is the call graph for this function:



6.71.1.16 char\* GetNewStringFromHTTPHeaderFieldPayload ( char\* request, unsigned int request\_length )

Here is the call graph for this function:



6.71.1.17 int ReducePathSlashes\_Inplace ( char \* filename )

Filenames may contain  $\frak {\frak {\frak$ 

# **Parameters**

Innut	string
mpat	cang

Return values

1=Success.	$\Lambda_{-}$	Fail	ııro
i=Success.	U=	гаш	ure

6.71.1.18 char\* RequestHTTPWebPage ( char\* hostname, unsigned int port, char\* filename, unsigned int max\_content )

A very basic http client for testing connections and maybe in the future make AmmarServers communicate with each other.

#### **Parameters**

Hostname	to connect to
Port	to connect to
Filename	to download
Maximum	size of response to carry

#### Return values

Pointer	to requested page,0=Failure

Here is the call graph for this function:



6.71.1.19 int seek\_blank\_char ( char \* input, char \* input\_end )

 $\textbf{6.71.1.20} \quad \text{int seek\_non\_blank\_char ( } \textbf{char} * \textbf{input}, \textbf{ char} * \textbf{input\_end )}$ 

6.71.1.21 unsigned int ServerThreads\_DropRootUID ( )

Drop Root UID, if we have one (and according to server\_configuration.h)

Here is the call graph for this function:



6.71.1.22 int setSocketTimeouts (int clientSock)

Enforce socket timeouts declared in server\_configuration.h and configuration files to socket.

#### **Parameters**

Socket	to change
--------	-----------

#### Return values

```
1=Success,0=Failure
```

Here is the call graph for this function:



6.71.1.23 int StripGETRequestQueryAndFragment ( char \* filename, char \* query, unsigned int max\_query\_length )

6.71.1.24 int StripHTMLCharacters\_Inplace ( char \* filename, int enable\_security )

HTML characters should be converted to plain c byte chars after we get them , this poses some security threats since this might allow "weird" bytes to get set that in conjunction with an overflow somewhere else might trick the server into executing ,.

#### **Parameters**

Input	string
Enforce	security that filters out possibly unwanted bytes!!, bytes larger than 255 are always filtered since the sec_byte can be also triggered by ZZ or any ascii value out of 0-F for ( see code )!

#### **Return values**

1=Success,0=Failure	

6.71.1.25 int StripVariableFromGETorPOSTString ( char \* input, char \* var\_id, char \* var\_val, unsigned int var\_val\_length )

TODO: A decent implementation here..!, input is like "idname=idvalue&idname2=idvalue2&idname3=idvalue3", var\_id is the value we are looking for var\_val is the payload which has space allocated as declared in var\_val\_length Here is the call graph for this function:

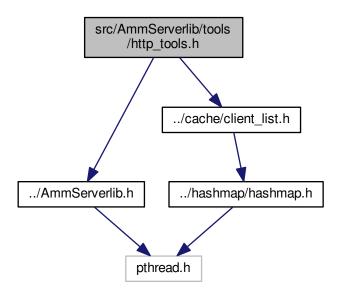


- 6.71.1.26 int stristr ( char \* str1CAPS, unsigned int str1\_length, char \* str2CAPS, unsigned int str2\_length, unsigned int \* pos\_found ) [inline]
- 6.71.1.27 int stristr2Caps ( char \* str1, unsigned int str1\_length, char \* str2CAPS, unsigned int str2\_length, unsigned int \* pos\_found ) [inline]
- 6.71.1.28 int strToUpcase ( char \* strTarget, char \* strSource, unsigned int strLength )
- 6.71.1.29 int trim\_last\_empty\_chars ( char \* input, unsigned int input\_length )

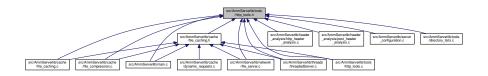
# 6.72 src/AmmServerlib/tools/http\_tools.h File Reference

A collection of tools required by the server and gathered here since they do a very specific job.

```
#include "../AmmServerlib.h"
#include "../cache/client_list.h"
Include dependency graph for http_tools.h:
```



This graph shows which files directly or indirectly include this file:



# **Typedefs**

typedef unsigned int contentType

#### **Enumerations**

enum contentTypeEnumerator {
 NO\_FILETYPE =0, RESERVED\_CTE\_VALUE, TEXT, IMAGE, AUDIO, VIDEO, EXECUTABLE, FOLDER }

### **Functions**

• unsigned int ServerThreads DropRootUID ()

Drop Root UID, if we have one (and according to server\_configuration.h)

char FileExistsAmmServ (char \*filename)

Check if file Exists.

char DirectoryExistsAmmServ (char \*dirpath)

Check if directory Exists.

int GetExtentionType (char \*theextension)

Convert an Extension Type to a contentTypeEnumerator.

int GetContentType (char \*filename, char \*contentType, unsigned int contentTypeLength)

Convert a filename to a contentType.

• int GetExtensionImage (char \*filename, char \*theimagepath, unsigned int theimagepath length)

Return template image for specific content type ( for directory listings etc )

- int FindIndexFile (struct AmmServer\_Instance \*instance, char \*webserver\_root, char \*directory, char \*indexfile)
- int StripGETRequestQueryAndFragment (char \*filename, char \*query, unsigned int max\_query\_length)
- int StripVariableFromGETorPOSTString (char \*input, char \*var\_id, char \*var\_val, unsigned int var\_val\_length)
- int GetDateString (char \*output, char \*label, unsigned int now, unsigned int dayofweek, unsigned int day, unsigned int month, unsigned int year, unsigned int hour, unsigned int minute, unsigned int second)
- int strToUpcase (char \*strTarget, char \*strSource, unsigned int strLength)
- int CheckHTTPHeaderCategoryAllCaps (char \*lineCAPS, unsigned int line\_length, char \*potential\_strCAPS, unsigned int \*payload start)
- int CheckHTTPHeaderCategory (char \*line, unsigned int line\_length, char \*potential\_strCAPS, unsigned int \*payload start)
- int trim last empty chars (char \*input, unsigned int input length)
- int seek\_non\_blank\_char (char \*input, char \*input\_end)
- int seek blank char (char \*input, char \*input end)
- unsigned int GetIntFromHTTPHeaderFieldPayload (char \*request, unsigned int request\_length)
- · char \* GetNewStringFromHTTPHeaderFieldPayload (char \*request, unsigned int request length)
- int encodeToBase64 (char \*src, unsigned s len, char \*dst, unsigned d len)

Convert a string to base64, required for the authorization tokens.

int StripHTMLCharacters\_Inplace (char \*filename, int enable\_security)

HTML characters should be converted to plain c byte chars after we get them , this poses some security threats since this might allow "weird" bytes to get set that in conjunction with an overflow somewhere else might trick the server into executing ,.

int ReducePathSlashes\_Inplace (char \*filename)

 $\textit{Filenames may contain ///// with an arbitrary number of slashes} \ , \ \textit{we convert them to a single slash} \ , \\$ 

• int FilenameStripperOk (char \*filename)

Strip filename and security check it.

char \* RequestHTTPWebPage (char \*hostname, unsigned int port, char \*filename, unsigned int max\_content)

A very basic http client for testing connections and maybe in the future make AmmarServers communicate with each other.

int freeString (char \*\*str)

Free C string and set it to 0.

• int setSocketTimeouts (int clientSock)

Enforce socket timeouts declared in server\_configuration.h and configuration files to socket.

• clientID findOutClientIDOfPeer (struct AmmServer\_Instance \*instance, int clientSock)

Tool that resolve a client socket to its IP, then uses it to try to clientList\_GetClientId and returns the id number.

# 6.72.1 Detailed Description

A collection of tools required by the server and gathered here since they do a very specific job.

**Author** 

Ammar Qammaz (AmmarkoV)

- 6.72.2 Typedef Documentation
- 6.72.2.1 typedef unsigned int contentType
- 6.72.3 Enumeration Type Documentation
- 6.72.3.1 enum contentTypeEnumerator

**Enumerator** 

NO\_FILETYPE

RESERVED\_CTE\_VALUE

**TEXT** 

**IMAGE** 

**AUDIO** 

**VIDEO** 

**EXECUTABLE** 

**FOLDER** 

# 6.72.4 Function Documentation

6.72.4.1 int CheckHTTPHeaderCategory ( char \* line, unsigned int line\_length, char \* potential\_strCAPS, unsigned int \* payload\_start )



6.72.4.2 int CheckHTTPHeaderCategoryAllCaps ( char \* lineCAPS, unsigned int line\_length, char \* potential\_strCAPS, unsigned int \* payload\_start )

Here is the call graph for this function:



6.72.4.3 char DirectoryExistsAmmServ ( char \* dirpath )

Check if directory Exists.

#### **Parameters**

Path	to directory

#### Return values

1=Exists,0=Does	not Exist

6.72.4.4 int encodeToBase64 ( char \* src, unsigned s\_len, char \* dst, unsigned d\_len )

Convert a string to base64, required for the authorization tokens.

# **Parameters**

Input	string
Input	<u> </u>
Output	string
Input	Maximum output length

# Return values

1=Success,0=Failure	

6.72.4.5 char FileExistsAmmServ ( char \* filename )

Check if file Exists.

**Parameters** 

Path	to file
------	---------

Return values

1=Exists,0=Does	not Exist
-----------------	-----------

6.72.4.6 int FilenameStripperOk ( char \* filename )

Strip filename and security check it.

#### **Parameters**

Pointer	to string pointer to be analyzed

#### Return values

1=Ok,0=Failed	

6.72.4.7 int FindIndexFile ( struct AmmServer\_Instance \* instance, char \* webserver\_root, char \* directory, char \* indexfile )

Here is the call graph for this function:



 $\textbf{6.72.4.8} \quad \textbf{clientID} \ \textbf{findOutClientIDOfPeer} \ ( \ \textbf{struct} \ \textbf{AmmServer\_Instance} * \textit{instance}, \ \textbf{int} \ \textit{clientSock} \ )$ 

Tool that resolve a client socket to its IP, then uses it to try to clientList\_GetClientId and returns the id number.

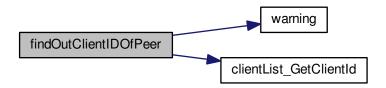
#### **Parameters**

An	AmmarServer instance	
client	socket	

#### Return values

ClientID	or ,0=Failure

Here is the call graph for this function:



6.72.4.9 int freeString ( char \*\* str )

Free C string and set it to 0.

#### **Parameters**

Pointer	to string pointer to be freed
---------	-------------------------------

#### **Return values**

1=Success,0=Failure	

6.72.4.10 int GetContentType ( char \* filename, char \* contentType, unsigned int contentTypeLength )

Convert a filename to a contentType.

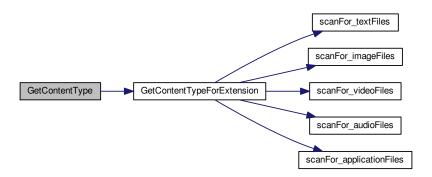
#### **Parameters**

String	with the filename we want to examine
Output	String with the contentType
Output	contentType length

# Return values

contentTypeEnumerator	

Here is the call graph for this function:



- 6.72.4.11 int GetDateString ( char \* output, char \* label, unsigned int now, unsigned int dayofweek, unsigned int day, unsigned int month, unsigned int year, unsigned int hour, unsigned int minute, unsigned int second )
- 6.72.4.12 int GetExtensionImage ( char \* filename, char \* theimagepath, unsigned int theimagepath\_length )

Return template image for specific content type ( for directory listings etc )

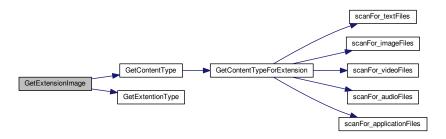
#### **Parameters**

Filename	of file
Path	to Image
Length	of path to Image

#### Return values

1=Exists,0=Does	not Exist

Here is the call graph for this function:



# 6.72.4.13 int GetExtentionType ( char \* theextension )

Convert an Extension Type to a contentTypeEnumerator.

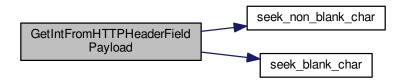
#### **Parameters**

String	with the extension type
--------	-------------------------

# Return values

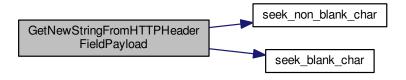
contentTypeEnumerator	

# 6.72.4.14 unsigned int GetIntFromHTTPHeaderFieldPayload ( char \* request, unsigned int request\_length )



6.72.4.15 char \* GetNewStringFromHTTPHeaderFieldPayload ( char \* request, unsigned int request\_length )

Here is the call graph for this function:



#### 6.72.4.16 int ReducePathSlashes\_Inplace ( char \* filename )

Filenames may contain ///// with an arbitrary number of slashes, we convert them to a single slash,.

#### **Parameters**

Input	string

#### Return values

1=Success,0=Failure	

6.72.4.17 char\* RequestHTTPWebPage ( char\* hostname, unsigned int port, char\* filename, unsigned int max\_content )

A very basic http client for testing connections and maybe in the future make AmmarServers communicate with each other.

### **Parameters**

Hostname	to connect to
Port	to connect to
Filename	to download
Maximum	size of response to carry

#### **Return values**

Pointer	to requested page,0=Failure



- 6.72.4.18 int seek\_blank\_char ( char \* input, char \* input\_end )
- 6.72.4.19 int seek\_non\_blank\_char ( char \* input, char \* input\_end )
- 6.72.4.20 unsigned int ServerThreads\_DropRootUID ( )

Drop Root UID, if we have one (and according to server\_configuration.h)

Here is the call graph for this function:



### 6.72.4.21 int setSocketTimeouts (int clientSock)

Enforce socket timeouts declared in server\_configuration.h and configuration files to socket.

#### **Parameters**

Socket	to change

### **Return values**

```
1=Success,0=Failure
```

Here is the call graph for this function:



- 6.72.4.22 int StripGETRequestQueryAndFragment ( char \* filename, char \* query, unsigned int max\_query\_length )
- 6.72.4.23 int StripHTMLCharacters\_Inplace ( char \* filename, int enable\_security )

HTML characters should be converted to plain c byte chars after we get them , this poses some security threats since this might allow "weird" bytes to get set that in conjunction with an overflow somewhere else might trick the server into executing ,.

#### **Parameters**

Input	string
Enforce	security that filters out possibly unwanted bytes!!, bytes larger than 255 are always filtered
	since the sec_byte can be also triggered by ZZ or any ascii value out of 0-F for ( see code )
	!

#### Return values

1=Success,0=Failure	

6.72.4.24 int StripVariableFromGETorPOSTString ( char \* input, char \* var\_id, char \* var\_val, unsigned int var\_val\_length )

TODO: A decent implementation here..!, input is like "idname=idvalue&idname2=idvalue2&idname3=idvalue3", var\_id is the value we are looking for var\_val is the payload which has space allocated as declared in var\_val\_length Here is the call graph for this function:



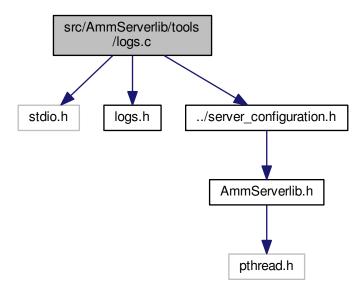
6.72.4.25 int strToUpcase ( char \* strTarget, char \* strSource, unsigned int strLength )

6.72.4.26 int trim\_last\_empty\_chars ( char \* input, unsigned int input\_length )

# 6.73 src/AmmServerlib/tools/logs.c File Reference

```
#include <stdio.h>
#include "logs.h"
#include "../server_configuration.h"
```

Include dependency graph for logs.c:



#### **Functions**

- void error (char \*msg)
   Log Function to output Errors.
- void warning (char \*msg)

Log Function to output warnings.

- int AccessLogAppend (char \*IP, char \*DateStr, char \*Request, unsigned int ResponseCode, unsigned long ResponseLength, char \*Location, char \*Useragent)
- int ErrorLogAppend (char \*IP, char \*DateStr, char \*Request, unsigned int ResponseCode, unsigned long ResponseLength, char \*Location, char \*Useragent)

# 6.73.1 Function Documentation

- 6.73.1.1 int AccessLogAppend ( char \* IP, char \* DateStr, char \* Request, unsigned int ResponseCode, unsigned long ResponseLength, char \* Location, char \* Useragent )
- 6.73.1.2 void error ( char \* msg )

Log Function to output Errors.

**Parameters** 

String with message To log

6.73.1.3 int ErrorLogAppend ( char \* IP, char \* DateStr, char \* Request, unsigned int ResponseCode, unsigned long ResponseLength, char \* Location, char \* Useragent )

6.73.1.4 void warning ( char \* msg )

Log Function to output warnings.

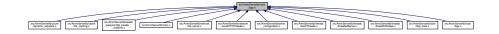
#### **Parameters**

String with message To log

# 6.74 src/AmmServerlib/tools/logs.h File Reference

Logging functions.

This graph shows which files directly or indirectly include this file:



#### **Macros**

- #define NORMAL "\033[0m"
- #define BLACK "\033[30m" /\* Black \*/
- #define RED "\033[31m" /\* Red \*/
- #define GREEN "\033[32m" /\* Green \*/
- #define YELLOW "\033[33m" /\* Yellow \*/
- #define BLUE "\033[34m" /\* Blue \*/
- #define MAGENTA "\033[35m" /\* Magenta \*/
- #define CYAN "\033[36m" /\* Cyan \*/
- #define WHITE "\033[37m" /\* White \*/
- #define BOLDBLACK "\033[1m\033[30m" /\* Bold Black \*/
- #define BOLDRED "\033[1m\033[31m" /\* Bold Red \*/
- #define BOLDGREEN "\033[1m\033[32m" /\* Bold Green \*/
- #define BOLDYELLOW "\033[1m\033[33m" /\* Bold Yellow \*/
- #define BOLDBLUE "\033[1m\033[34m" /\* Bold Blue \*/
- #define BOLDMAGENTA "\033[1m\033[35m" /\* Bold Magenta \*/
- #define BOLDCYAN "\033[1m\033[36m" /\* Bold Cyan \*/
- #define BOLDWHITE "\033[1m\033[37m" /\* Bold White \*/

# **Functions**

void error (char \*msg)

Log Function to output Errors.

void warning (char \*msg)

Log Function to output warnings.

- int AccessLogAppend (char \*IP, char \*DateStr, char \*Request, unsigned int ResponseCode, unsigned long ResponseLength, char \*Location, char \*Useragent)
- int ErrorLogAppend (char \*IP, char \*DateStr, char \*Request, unsigned int ResponseCode, unsigned long ResponseLength, char \*Location, char \*Useragent)

# 6.74.1 Detailed Description

Logging functions.

Author

Ammar Qammaz (AmmarkoV)

```
6.74.2 Macro Definition Documentation
6.74.2.1 #define BLACK "\033[30m" /* Black */
6.74.2.2 #define BLUE "\033[34m" /* Blue */
6.74.2.3 #define BOLDBLACK "\033[1m\033[30m" /* Bold Black */
6.74.2.4 #define BOLDBLUE "\033[1m\033[34m" /* Bold Blue */
6.74.2.5 #define BOLDCYAN "\033[1m\033[36m" /* Bold Cyan */
6.74.2.6 #define BOLDGREEN "\033[1m\033[32m" /* Bold Green */
6.74.2.7 #define BOLDMAGENTA "\033[1m\033[35m" /* Bold Magenta */
6.74.2.8 #define BOLDRED "\033[1m\033[31m" /* Bold Red */
6.74.2.9 #define BOLDWHITE "\033[1m\033[37m" /* Bold White */
6.74.2.10 #define BOLDYELLOW "\033[1m\033[33m" /* Bold Yellow */
6.74.2.11 #define CYAN "\033[36m" /* Cyan */
6.74.2.12 #define GREEN "\033[32m" /* Green */
6.74.2.13 #define MAGENTA "\033[35m" /* Magenta */
6.74.2.14 #define NORMAL "\033[0m"
6.74.2.15 #define RED "\033[31m" /* Red */
6.74.2.16 #define WHITE "\033[37m" /* White */
6.74.2.17 #define YELLOW "\033[33m" /* Yellow */
6.74.3 Function Documentation
6.74.3.1 int AccessLogAppend ( char * IP, char * DateStr, char * Request, unsigned int ResponseCode, unsigned long
         ResponseLength, char * Location, char * Useragent )
6.74.3.2 void error ( char * msg )
Log Function to output Errors.
Parameters
             String | with message To log
```

6.74.3.3 int ErrorLogAppend ( char \* IP, char \* DateStr, char \* Request, unsigned int ResponseCode, unsigned long

```
6.74.3.4 void warning ( char * msg )
```

Log Function to output warnings.

ResponseLength, char \* Location, char \* Useragent )

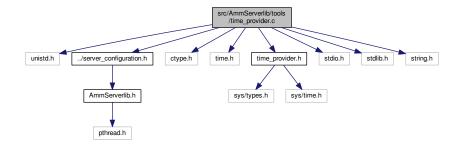
#### **Parameters**

String with message To log

# 6.75 src/AmmServerlib/tools/time\_provider.c File Reference

```
#include <unistd.h>
#include "../server_configuration.h"
#include <ctype.h>
#include <time.h>
#include "time_provider.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

Include dependency graph for time provider.c:



# **Functions**

unsigned long GetTickCountAmmServ ()

GetTickCount like call for functions wanting to get monotonic values in milliseconds.

• int GetDateString (char \*output, char \*label, unsigned int now, unsigned int dayofweek, unsigned int day, unsigned int month, unsigned int year, unsigned int hour, unsigned int minute, unsigned int second)

Get a string back with date and time.

int start\_timer (struct time\_snap \*val)

Start a timer using a time\_snap structure.

• unsigned long end\_timer (struct time\_snap \*val)

End a started timer and get back the results.

### **Variables**

```
const char * days [] = {"Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"}
```

 $\bullet \ \ const \ char * months \ [] = \{"Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec"\}$ 

# 6.75.1 Function Documentation

6.75.1.1 unsigned long end\_timer ( struct time\_snap \* val )

End a started timer and get back the results.

#### **Parameters**

time_snap	structure that holds the timer data

#### **Return values**

Elapsed	time since start_timer, needs to be divided by 1000 to get msecs, and by
	1000000 to get seconds

6.75.1.2 int GetDateString ( char \* output, char \* label, unsigned int now, unsigned int dayofweek, unsigned int day, unsigned int month, unsigned int year, unsigned int hour, unsigned int minute, unsigned int second )

Get a string back with date and time.

#### **Parameters**

Pointer	to where Output String should be stored
Pointer	to Label String
Flag	to control if we want to override values with the current time
Unsigned	Integer Day of Week Value
Unsigned	Integer Day
Unsigned	Integer Month
Unsigned	Integer Year
Unsigned	Integer Hour
Unsigned	Integer Minute
Unsigned	Integer Second

#### **Return values**

1=Success,0=Failure	

# 6.75.1.3 unsigned long GetTickCountAmmServ ( )

GetTickCount like call for functions wanting to get monotonic values in milliseconds.

#### **Return values**

Milliseconds	

6.75.1.4 int start\_timer ( struct time\_snap \* val )

Start a timer using a time\_snap structure.

### **Parameters**

time_snap	structure that holds the timer data

### Return values

1=Success,0=Failure	

# 6.75.2 Variable Documentation

```
6.75.2.1 const char* days[] = {"Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"}
```

6.75.2.2 const char\* months[] = {"Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"}

# 6.76 src/AmmServerlib/tools/time\_provider.h File Reference

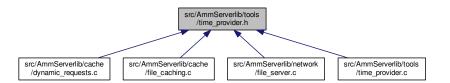
#### Timer functions.

#include <sys/types.h>
#include <sys/time.h>
Include dependency graph for time provider.h:

src/AmmServerlib/tools
/time\_provider.h

sys/types.h
sys/time.h

This graph shows which files directly or indirectly include this file:



# **Data Structures**

• struct time\_snap

# **Functions**

• unsigned long GetTickCountAmmServ ()

GetTickCount like call for functions wanting to get monotonic values in milliseconds.

• int GetDateString (char \*output, char \*label, unsigned int now, unsigned int dayofweek, unsigned int day, unsigned int month, unsigned int year, unsigned int hour, unsigned int minute, unsigned int second)

Get a string back with date and time.

• int start\_timer (struct time\_snap \*val)

Start a timer using a time\_snap structure.

• unsigned long end\_timer (struct time\_snap \*val)

End a started timer and get back the results.

# 6.76.1 Detailed Description

Timer functions.

**Author** 

Ammar Qammaz (AmmarkoV)

# 6.76.2 Function Documentation

6.76.2.1 unsigned long end\_timer ( struct time\_snap \* val )

End a started timer and get back the results.

#### **Parameters**

time_snap	structure that holds the timer data

#### Return values

Elapsed	time since start_timer, needs to be divided by 1000 to get msecs, and by
	1000000 to get seconds

6.76.2.2 int GetDateString ( char \* output, char \* label, unsigned int now, unsigned int dayofweek, unsigned int day, unsigned int month, unsigned int year, unsigned int hour, unsigned int minute, unsigned int second )

Get a string back with date and time.

#### **Parameters**

Pointer	to where Output String should be stored
Pointer	to Label String
Flag	to control if we want to override values with the current time
Unsigned	Integer Day of Week Value
Unsigned	Integer Day
Unsigned	Integer Month
Unsigned	Integer Year
Unsigned	Integer Hour
Unsigned	Integer Minute
Unsigned	Integer Second

### Return values

1=Success,0=Failure
---------------------

6.76.2.3 unsigned long GetTickCountAmmServ ( )

GetTickCount like call for functions wanting to get monotonic values in milliseconds.

#### **Return values**

Milliseconds	

6.76.2.4 int start\_timer ( struct time\_snap \* val )

Start a timer using a time\_snap structure.

#### **Parameters**

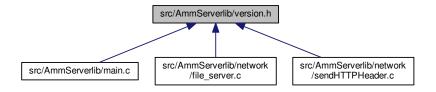
time_snap	structure that holds the timer data
-----------	-------------------------------------

#### Return values

```
1=Success,0=Failure
```

# 6.77 src/AmmServerlib/version.h File Reference

This graph shows which files directly or indirectly include this file:



#### **Macros**

- #define RC\_FILEVERSION 0,27,124,537
- #define RC\_FILEVERSION\_STRING "0, 27, 124, 537\0"

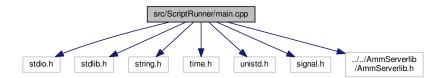
# 6.77.1 Macro Definition Documentation

- 6.77.1.1 #define RC\_FILEVERSION 0,27,124,537
- 6.77.1.2 #define RC\_FILEVERSION\_STRING "0, 27, 124, 537\0"

# 6.78 src/ScriptRunner/main.cpp File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <unistd.h>
#include <signal.h>
#include "../../AmmServerlib/AmmServerlib.h"
```

Include dependency graph for main.cpp:



#### **Macros**

- #define MAX BINDING PORT 65534
- #define ENABLE PASSWORD PROTECTION 0
- #define ENABLE CHAT BOX 0
- #define MAX COMMAND SIZE 2048
- #define DEFAULT BINDING PORT 8080
- #define ADMIN BINDING PORT 8082
- #define ENABLE\_ADMIN\_PAGE 0

#### **Functions**

- void replaceChar (char \*input, char findChar, char replaceWith)
- void \* prepare index content callback (struct AmmServer DynamicRequest \*rqst)
- int getBackCommandLine (char \*command, char \*what2GetBack, unsigned int what2GetBackMaxSize)
- void \* prepare stats content callback (struct AmmServer DynamicRequest \*rgst)
- void \* prepare\_base\_image (struct AmmServer\_DynamicRequest \*rqst)
- void \* prepare\_top\_image (struct AmmServer\_DynamicRequest \*rqst)
- void joystickExecute (float x, float y)
- void execute (char \*command, char \*param)
- void \* store\_new\_configuration\_callback (struct AmmServer\_DynamicRequest \*rqst)
- void \* prepare\_form\_content\_callback (struct AmmServer\_DynamicRequest \*rqst)
- int init\_dynamic\_content ()
- · void close\_dynamic\_content ()
- void termination\_handler (int signum)
- int main (int argc, char \*argv[])

#### **Variables**

- char admin root [MAX FILE PATH] = "admin html/"
- char webserver\_root [MAX\_FILE\_PATH] ="public\_html/"
- char templates\_root [MAX\_FILE\_PATH] ="public\_html/templates/"
- char \* page =0
- unsigned int pageLength =0
- struct AmmServer Instance \* default server =0
- struct AmmServer\_Instance \* admin\_server =0
- · struct
  - AmmServer\_RequestOverride\_Context GET\_override ={{0}}
- struct AmmServer\_RH\_Context indexPage ={0}
- struct AmmServer RH Context settings ={0}
- struct AmmServer\_RH\_Context stats ={0}

- struct AmmServer\_RH\_Context form ={0}
- struct AmmServer\_RH\_Context chatbox ={0}
- struct AmmServer\_RH\_Context base\_image ={0}
- struct AmmServer\_RH\_Context top\_image ={0}
- struct AmmServer\_RH\_Context random\_chars ={0}

# 6.78.1 Macro Definition Documentation

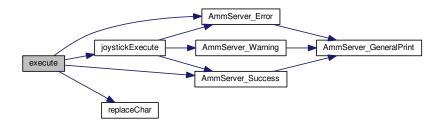
- 6.78.1.1 #define ADMIN\_BINDING\_PORT 8082
- 6.78.1.2 #define DEFAULT\_BINDING\_PORT 8080
- 6.78.1.3 #define ENABLE\_ADMIN\_PAGE 0
- 6.78.1.4 #define ENABLE\_CHAT\_BOX 0
- 6.78.1.5 #define ENABLE\_PASSWORD\_PROTECTION 0
- 6.78.1.6 #define MAX\_BINDING\_PORT 65534
- 6.78.1.7 #define MAX\_COMMAND\_SIZE 2048
- 6.78.2 Function Documentation
- 6.78.2.1 void close\_dynamic\_content ( )

Here is the call graph for this function:



6.78.2.2 void execute ( char \* command, char \* param )

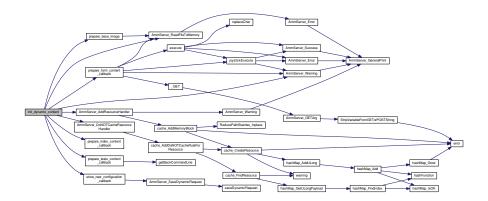
# bin/bash -c "



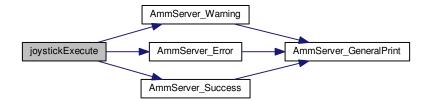
6.78.2.3 int getBackCommandLine ( char \* command, char \* what2GetBack, unsigned int what2GetBackMaxSize )

6.78.2.4 int init\_dynamic\_content ( )

Here is the call graph for this function:

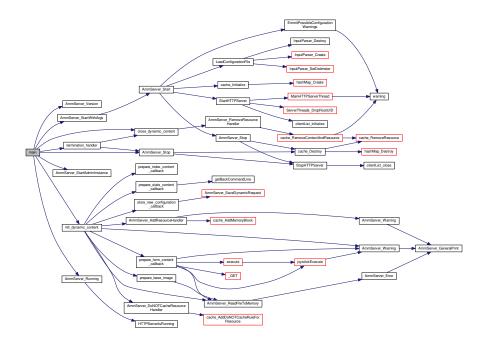


# 6.78.2.5 void joystickExecute ( float x, float y)



# 6.78.2.6 int main ( int argc, char \* argv[] )

Here is the call graph for this function:

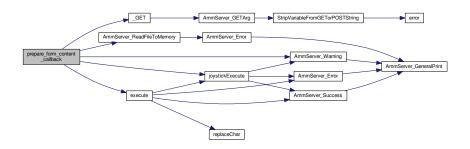


# 6.78.2.7 void\* prepare\_base\_image ( struct AmmServer\_DynamicRequest \* rqst )

Here is the call graph for this function:



# 6.78.2.8 void\* prepare\_form\_content\_callback ( struct AmmServer\_DynamicRequest \* rqst )



6.78.2.9 void\* prepare\_index\_content\_callback ( struct AmmServer\_DynamicRequest \* rqst )

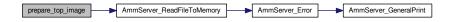
6.78.2.10 void\* prepare\_stats\_content\_callback ( struct AmmServer\_DynamicRequest \* rqst )

Here is the call graph for this function:



6.78.2.11 void\* prepare\_top\_image ( struct AmmServer\_DynamicRequest \* rqst )

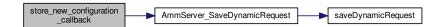
Here is the call graph for this function:



6.78.2.12 void replaceChar ( char \* input, char findChar, char replaceWith )

6.78.2.13 void\* store\_new\_configuration\_callback ( struct AmmServer\_DynamicRequest \* rqst )

Here is the call graph for this function:



6.78.2.14 void termination\_handler ( int signum )

Dynamic content code ..! END -----

Here is the call graph for this function:



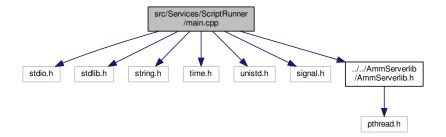
326 File Documentation

```
6.78.3 Variable Documentation
6.78.3.1 char admin_root[MAX_FILE_PATH] = "admin_html/"
6.78.3.2 struct AmmServer_Instance* admin_server =0
6.78.3.3 struct AmmServer_RH_Context base_image ={0}
6.78.3.4 struct AmmServer_RH_Context chatbox ={0}
6.78.3.5 struct AmmServer_Instance* default_server =0
Dynamic content code ..! START!
6.78.3.6 struct AmmServer_RH_Context form ={0}
6.78.3.7 struct AmmServer_RequestOverride_Context GET_override ={{0}}
6.78.3.8 struct AmmServer_RH_Context indexPage ={0}
6.78.3.9 char* page =0
6.78.3.10 unsigned int pageLength =0
6.78.3.11 struct AmmServer_RH_Context random_chars ={0}
6.78.3.12 struct AmmServer_RH_Context settings ={0}
6.78.3.13 struct AmmServer_RH_Context stats ={0}
6.78.3.14 char templates_root[MAX_FILE_PATH] = "public_html/templates/"
6.78.3.15 struct AmmServer_RH_Context top_image ={0}
6.78.3.16 char webserver_root[MAX_FILE_PATH] = "public_html/"
6.79 src/Services/ScriptRunner/main.cpp File Reference
#include <stdio.h>
```

#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <unistd.h>
#include <signal.h>

#include "../../AmmServerlib/AmmServerlib.h"

Include dependency graph for main.cpp:



#### **Macros**

- #define MAX BINDING PORT 65534
- #define ENABLE PASSWORD PROTECTION 0
- #define ENABLE\_CHAT\_BOX 0
- #define MAX COMMAND SIZE 2048
- #define DEFAULT\_BINDING\_PORT 8080
- #define ADMIN BINDING PORT 8082
- #define ENABLE ADMIN PAGE 0

#### **Functions**

- char FileExistsTest (char \*filename)
- char EraseFile (char \*filename)
- unsigned int StringIsHTMLSafe (char \*str)
- void replaceChar (char \*input, char findChar, char replaceWith)
- void \* prepare index content callback (struct AmmServer DynamicRequest \*rqst)
- int getBackCommandLine (char \*command, char \*what2GetBack, unsigned int what2GetBackMaxSize)
- void \* prepare\_stats\_content\_callback (struct AmmServer\_DynamicRequest \*rqst)
- void \* prepare\_base\_image (struct AmmServer\_DynamicRequest \*rqst)
- void \* prepare\_top\_image (struct AmmServer\_DynamicRequest \*rqst)
- void joystickExecute (float x, float y)
- void execute (char \*command, char \*param)
- void \* store\_new\_configuration\_callback (struct AmmServer\_DynamicRequest \*rqst)
- void \* prepare\_form\_content\_callback (struct AmmServer\_DynamicRequest \*rqst)
- int init\_dynamic\_content ()
- void close\_dynamic\_content ()
- void termination\_handler (int signum)
- int main (int argc, char \*argv[])

#### **Variables**

- char admin\_root [MAX\_FILE\_PATH] = "admin\_html/"
- char webserver\_root [MAX\_FILE\_PATH] ="public\_html/"
- char templates\_root [MAX\_FILE\_PATH] ="public\_html/templates/"
- char \* page =0
- unsigned int pageLength =0
- struct AmmServer\_Instance \* default\_server =0

328 File Documentation

- struct AmmServer\_Instance \* admin\_server =0
- struct AmmServer\_RequestOverride\_Context GET\_override ={{0}}
- struct AmmServer\_RH\_Context indexPage ={0}
- struct AmmServer\_RH\_Context settings ={0}
- struct AmmServer\_RH\_Context stats ={0}
- struct AmmServer\_RH\_Context form ={0}
- struct AmmServer\_RH\_Context chatbox ={0}
- struct AmmServer\_RH\_Context base\_image ={0}
- struct AmmServer\_RH\_Context top\_image ={0}
- struct AmmServer\_RH\_Context random\_chars ={0}

#### 6.79.1 Macro Definition Documentation

- 6.79.1.1 #define ADMIN\_BINDING\_PORT 8082
- 6.79.1.2 #define DEFAULT\_BINDING\_PORT 8080
- 6.79.1.3 #define ENABLE\_ADMIN\_PAGE 0
- 6.79.1.4 #define ENABLE\_CHAT\_BOX 0
- 6.79.1.5 #define ENABLE\_PASSWORD\_PROTECTION 0
- 6.79.1.6 #define MAX\_BINDING\_PORT 65534
- 6.79.1.7 #define MAX\_COMMAND\_SIZE 2048

#### 6.79.2 Function Documentation

6.79.2.1 void close\_dynamic\_content ( )

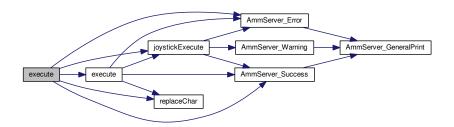
Here is the call graph for this function:



- 6.79.2.2 char EraseFile ( char \* filename )
- 6.79.2.3 void execute ( char \* command, char \* param )

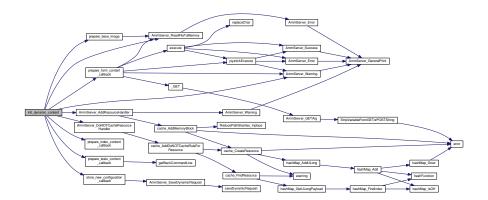
bin/bash -c "

Here is the call graph for this function:



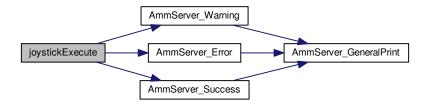
- 6.79.2.4 char FileExistsTest ( char \* filename )
- 6.79.2.5 int getBackCommandLine ( char \* command, char \* what2GetBack, unsigned int what2GetBackMaxSize )
- 6.79.2.6 int init\_dynamic\_content ( )

Here is the call graph for this function:



## 6.79.2.7 void joystickExecute (float x, float y)

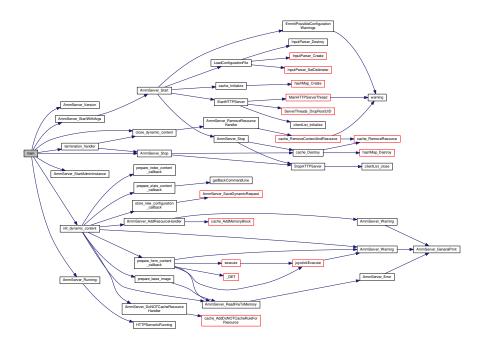
Here is the call graph for this function:



330 File Documentation

## 6.79.2.8 int main ( int argc, char \* argv[] )

Here is the call graph for this function:



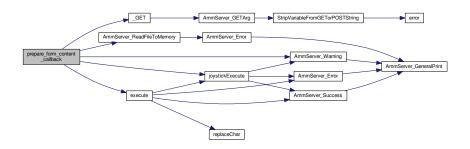
## 6.79.2.9 void\* prepare\_base\_image ( struct AmmServer\_DynamicRequest \* rqst )

Here is the call graph for this function:



## 6.79.2.10 void\* prepare\_form\_content\_callback ( struct AmmServer\_DynamicRequest \* rqst )

Here is the call graph for this function:



6.79.2.11 void\* prepare\_index\_content\_callback ( struct AmmServer\_DynamicRequest \* rqst )

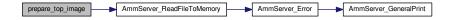
6.79.2.12 void\* prepare\_stats\_content\_callback ( struct AmmServer\_DynamicRequest \* rqst )

Here is the call graph for this function:



6.79.2.13 void\* prepare\_top\_image ( struct AmmServer\_DynamicRequest \* rqst )

Here is the call graph for this function:



6.79.2.14 void replaceChar ( char \* input, char findChar, char replaceWith )

6.79.2.15 void\* store\_new\_configuration\_callback ( struct AmmServer\_DynamicRequest \* rqst )

Here is the call graph for this function:



6.79.2.16 unsigned int StringlsHTMLSafe ( char \* str )

6.79.2.17 void termination\_handler ( int signum )

Dynamic content code ..! END -----

332 File Documentation

Here is the call graph for this function:



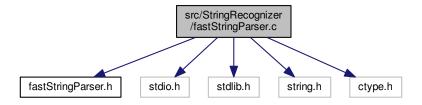
```
6.79.3 Variable Documentation
6.79.3.1 char admin_root[MAX_FILE_PATH] = "admin_html/"
6.79.3.2 struct AmmServer_Instance* admin_server =0
6.79.3.3 struct AmmServer_RH_Context base_image ={0}
6.79.3.4 struct AmmServer RH Context chatbox ={0}
6.79.3.5 struct AmmServer_Instance* default_server =0
Dynamic content code ..! START!
6.79.3.6 struct AmmServer_RH_Context form ={0}
6.79.3.7 struct AmmServer_RequestOverride_Context GET_override ={{0}}
6.79.3.8 struct AmmServer_RH_Context indexPage ={0}
6.79.3.9 char* page =0
6.79.3.10 unsigned int pageLength =0
6.79.3.11 struct AmmServer_RH_Context random_chars ={0}
6.79.3.12 struct AmmServer_RH_Context settings ={0}
6.79.3.13 struct AmmServer_RH_Context stats ={0}
6.79.3.14 char templates_root[MAX_FILE_PATH] = "public_html/templates/"
6.79.3.15 struct AmmServer_RH_Context top_image ={0}
```

# 6.80 src/StringRecognizer/fastStringParser.c File Reference

6.79.3.16 char webserver\_root[MAX\_FILE\_PATH] = "public\_html/"

```
#include "fastStringParser.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
```

Include dependency graph for fastStringParser.c:



#### **Macros**

- #define MAXIMUM LINE LENGTH 1024
- #define MAXIMUM LEVELS 123
- #define ACTIVATED\_LEVELS 3

#### **Functions**

- void convertTo ENUM ID (char \*sPtr)
- int fastStringParser\_addString (struct fastStringParser \*fsp, char \*str)
- struct fastStringParser \* fastStringParser initialize (unsigned int totalStrings)
- int fastStringParser\_hasStringsWithNConsecutiveChars (struct fastStringParser \*fsp, unsigned int \*res-StringResultIndex, char \*Sequence, unsigned int seqLength)
- unsigned int fastStringParser\_countStringsForNextChar (struct fastStringParser \*fsp, unsigned int \*resString-ResultIndex, char \*Sequence, unsigned int seqLength)
- void addLevelSpaces (FILE \*fp, unsigned int level)
- int printlfAllPossibleStrings (FILE \*fp, struct fastStringParser \*fsp, char \*Sequence, unsigned int seqLength)
- int printAllEnumeratorItems (FILE \*fp, struct fastStringParser \*fsp, char \*functionName)
- int recursiveTraverser (FILE \*fp, struct fastStringParser \*fsp, char \*functionName, char \*cArray, unsigned int level)
- int export\_C\_Scanner (struct fastStringParser \*fsp, char \*functionName)

  Export a C Scanner source code.
- struct fastStringParser \* fastSTringParser\_createRulesFromFile (char \*filename, unsigned int totalStrings)

  Read a file and create C files that parse the input.
- int fastStringParser\_close ()

#### **Variables**

- struct fastStringParser \* fspHTTPHeader = 0
- char acceptedChars [] ="ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789-\_"
- 6.80.1 Macro Definition Documentation
- 6.80.1.1 #define ACTIVATED\_LEVELS 3
- 6.80.1.2 #define MAXIMUM\_LEVELS 123
- 6.80.1.3 #define MAXIMUM\_LINE\_LENGTH 1024

334 File Documentation

## 6.80.2 Function Documentation

6.80.2.1 void addLevelSpaces (FILE \* fp, unsigned int level )

6.80.2.2 void convertTo\_ENUM\_ID ( char \* sPtr ) [inline]

6.80.2.3 int export\_C\_Scanner ( struct fastStringParser \* fsp, char \* filename )

Export a C Scanner source code.

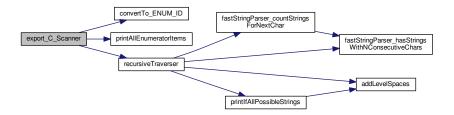
#### **Parameters**

Structure	to hold all the intermediate state
Name	of the current function

#### **Return values**

1 Curana O Failura	
1=Success.0=Failure	
. 545555,5 . 4	

Here is the call graph for this function:



6.80.2.4 int fastStringParser\_addString ( struct fastStringParser \* fsp, char \* str )

Here is the call graph for this function:



6.80.2.5 int fastStringParser\_close ( )

6.80.2.6 unsigned int fastStringParser\_countStringsForNextChar ( struct fastStringParser \* fsp, unsigned int \* resStringResultIndex, char \* Sequence, unsigned int seqLength )

Here is the call graph for this function:



6.80.2.7 struct fastStringParser\* fastSTringParser\_createRulesFromFile ( char \* filename, unsigned int totalStrings )

Read a file and create C files that parse the input.

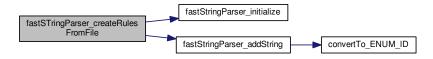
#### **Parameters**

Filename	of the current function
Total	Number of Strings

#### Return values

fastStringParser	context,0=Failure

Here is the call graph for this function:



- 6.80.2.8 int fastStringParser\_hasStringsWithNConsecutiveChars ( struct fastStringParser \* fsp, unsigned int \* resStringResultIndex, char \* Sequence, unsigned int seqLength )
- 6.80.2.9 struct fastStringParser\* fastStringParser\_initialize (unsigned int totalStrings)
- 6.80.2.10 int printAllEnumeratorItems (FILE \* fp, struct fastStringParser \* fsp, char \* functionName )

336 File Documentation

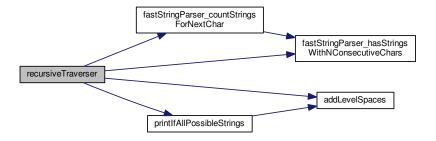
6.80.2.11 int printlfAllPossibleStrings (FILE \* fp, struct fastStringParser \* fsp, char \* Sequence, unsigned int seqLength)

Here is the call graph for this function:



6.80.2.12 int recursiveTraverser ( FILE \* fp, struct fastStringParser \* fsp, char \* functionName, char \* cArray, unsigned int level )

Here is the call graph for this function:



## 6.80.3 Variable Documentation

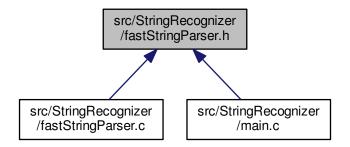
6.80.3.1 char acceptedChars[] = "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789-\_"

6.80.3.2 struct fastStringParser\* fspHTTPHeader = 0

# 6.81 src/StringRecognizer/fastStringParser.h File Reference

A tool that converts a file with words ( each word on a new line ) to C code ( see automata ) for fast string checking.

This graph shows which files directly or indirectly include this file:



## **Data Structures**

- struct fspString
  - Internal Structure to hold a string and its id for further processing.
- · struct fastStringParser

Internal Structure that holds all the string parser context.

#### **Functions**

- int export\_C\_Scanner (struct fastStringParser \*fsp, char \*filename)
  - Export a C Scanner source code.
- struct fastStringParser \* fastSTringParser\_createRulesFromFile (char \*filename, unsigned int totalStrings)

  Read a file and create C files that parse the input.

## 6.81.1 Detailed Description

A tool that converts a file with words ( each word on a new line ) to C code ( see automata ) for fast string checking.

Author

Ammar Qammaz (AmmarkoV)

#### 6.81.2 Function Documentation

6.81.2.1 int export\_C\_Scanner ( struct fastStringParser \* fsp, char \* filename )

Export a C Scanner source code.

**Parameters** 

Structure	to hold all the intermediate state

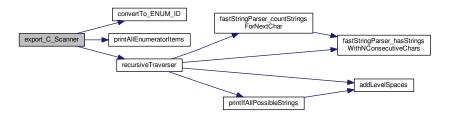
338 File Documentation

Name	e of the current function	

Return values

```
1=Success,0=Failure
```

Here is the call graph for this function:



6.81.2.2 struct fastStringParser\* fastSTringParser\_createRulesFromFile ( char \* filename, unsigned int totalStrings )

Read a file and create C files that parse the input.

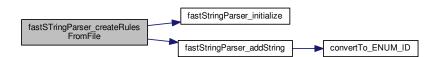
## **Parameters**

Filename	of the current function
Total	Number of Strings

## Return values

fastStringParser	context,0=Failure

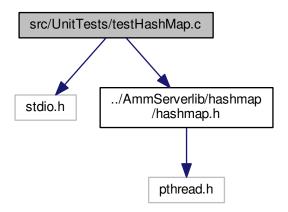
Here is the call graph for this function:



# 6.82 src/UnitTests/testHashMap.c File Reference

#include <stdio.h>
#include "../AmmServerlib/hashmap/hashmap.h"

Include dependency graph for testHashMap.c:



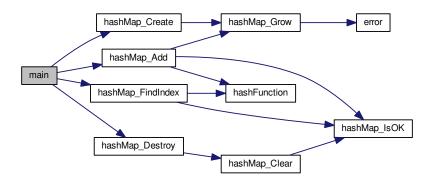
## **Functions**

• int main (int argc, char \*argv[])

## 6.82.1 Function Documentation

6.82.1.1 int main ( int argc, char \* argv[] )

Here is the call graph for this function:



# Index

AUDIOFILES_END_OF_ITEMS
audioFiles.h, 250
AUDIOFILES MID
audioFiles.h, 250
AUDIOFILES MP3
audioFiles.h, 250
AUDIOFILES OGG
audioFiles.h, 250
AUDIOFILES VOC
audioFiles.h, 250
AUDIOFILES WAV
audioFiles.h, 250
ABS
img_warp.c, 101
ABSDIFF
img_warp.c, 101
ACTIVATED LEVELS
<del>-</del>
fastStringParser.c, 333
ADMIN_BINDING_PORT
ScriptRunner/main.cpp, 322
Services/AmmarServer/main.c, 76
Services/ScriptRunner/main.cpp, 328
acceptedChars
fastStringParser.c, 336
AccessLog
server_configuration.c, 235
server_configuration.h, 245
AccessLogAppend
logs.c, 312
logs.h, 315
AccessLogEnable
server_configuration.c, 235
server_configuration.h, 245
Add_MyURL
Services/MyURL/main.c, 87
addLevelSpaces
fastStringParser.c, 334
admin_root
ScriptRunner/main.cpp, 326
Services/AmmarServer/main.c, 78
Services/GeoPosShare/main.c, 81
Services/ScriptRunner/main.cpp, 332
admin_server
ScriptRunner/main.cpp, 326
Services/AmmarServer/main.c, 78
Services/ScriptRunner/main.cpp, 332
allocateLinksIfNeeded
Services/MyURL/main.c, 88
allocated links

Services/MyURL/main.c, 94	AmmCaptcha/main.c, 48
AmmServerlib.h	AmmCaptcha_loadDictionary
AMMINF_ACTIVE_CLIENTS, 114	AmmCaptcha/main.c, 48
AMMINF_ACTIVE_THREADS, 114	AmmServInfos
AMMSET_PASSWORD_PROTECTION, 114	AmmServerlib.h, 113
AMMSET PASSWORD STR, 114	AmmServSettings
	AmmServerlib.h, 114
AMMSET_TEST, 114	
AMMSET_TESTSTR, 114	AmmServStrSettings
AMMSET_USERNAME_STR, 114	AmmServerlib.h, 114
BAD, 114	AmmServer_AddRequestHandler
CONNECT, 114	AmmServerlib.h, 115
DELETE, 114	AmmServerlib/main.c, 56
DIFFERENT_PAGE_FOR_EACH_CLIENT, 114	AmmServer_AddResourceHandler
GET, 114	AmmServerlib.h, 116
HEAD, 114	AmmServerlib/main.c, 57
NONE, 114	AmmServer_CheckIfHeaderBinaryAreTheSame
OPTIONS, 114	AmmServerlib.h, 116
PATCH, 114	AmmServerlib/main.c, 57
POST, 114	AmmServer_DirectoryExists
PUT, 114	AmmServerlib.h, 117
SAME_PAGE_FOR_ALL_CLIENTS, 114	AmmServerlib/main.c, 58
TRACE, 114	AmmServer_DoNOTCacheResource
AmmCaptcha.h	AmmServerlib.h, 117
AmmCaptcha_destroy, 43	AmmServerlib/main.c, 58
AmmCaptcha_getCaptchaFrame, 43	AmmServer_DoNOTCacheResourceHandler
AmmCaptcha_initialize, 44	AmmServerlib.h, 117
AmmCaptcha_isReplyCorrect, 44	AmmServerlib/main.c, 58
testAmmCaptcha, 44	AmmServer_DynamicRequest, 13
AmmCaptcha/AmmCaptchaTester/main.c	clientID, 13
main, 45	compressedContent, 13
AmmCaptcha/main.c	compressedContentSize, 13
AmmCaptcha_copyCaptchaJPEGImageWithCopy,	content, 13
47	contentSize, 13
AmmCaptcha_destroy, 47	GET_request, 13
AmmCaptcha_getCaptchaFrame, 47	GET_request_length, 14
AmmCaptcha_initialize, 48	headerResponse, 14
AmmCaptcha_isReplyCorrect, 48	MAXcompressedContentSize, 14
AmmCaptcha_isrteplyContect, 40  AmmCaptcha_loadDictionary, 48	•
• —	MAXcontentSize, 14
captchaStrings, 50 convertExternalIDToInternal, 49	POST_request, 14
	POST_request_length, 14
fontRAW, 50	AmmServer_EraseFile
fontX, 50	AmmServerlib.h, 118
fontY, 50	AmmServerlib/main.c, 59
RenderString, 49	AmmServer_Error
testAmmCaptcha, 49	AmmServerlib.h, 118
AmmCaptcha_copyCaptchaJPEGImageWithCopy	AmmServerlib/main.c, 59
AmmCaptcha/main.c, 47	AmmServer_ExecuteCommandLine
AmmCaptcha_destroy	AmmServerlib.h, 118
AmmCaptcha.h, 43	AmmServerlib/main.c, 59
AmmCaptcha/main.c, 47	AmmServer_FILES
AmmCaptcha_getCaptchaFrame	AmmServerlib.h, 119
AmmCaptcha.h, 43	AmmServerlib/main.c, 60
AmmCaptcha/main.c, 47	AmmServer_FileExists
AmmCaptcha_initialize	AmmServerlib.h, 119
AmmCaptcha.h, 44	AmmServerlib/main.c, 60
AmmCaptcha/main.c, 48	AmmServer_GETArg
AmmCaptcha_isReplyCorrect	AmmServerlib.h, 119
AmmCaptcha.h, 44	AmmServerlib/main.c, 60

AmmServer_GeneralPrint	AmmServerlib.h, 121
AmmServerlib/main.c, 60	AmmServerlib/main.c, 65
AmmServer_GetInfo	AmmServer_RemoveResourceHandler
AmmServerlib.h, 120	AmmServerlib.h, 123
AmmServerlib/main.c, 62	AmmServerlib/main.c, 65
AmmServer_GetIntSettingValue	AmmServer_ReplaceVarInMemoryFile
AmmServerlib.h, 120	AmmServerlib.h, 123
AmmServerlib/main.c, 62	AmmServerlib/main.c, 66
AmmServer_GetStrSettingValue	AmmServer_RequestOverride_Context, 17
AmmServerlib.h, 120	request, 17
AmmServerlib/main.c, 62	request_override_callback, 17
AmmServer_GlobalTerminationHandler	requestHeader, 17
AmmServerlib/main.c, 63	AmmServer_Running
AmmServer_Instance, 14	AmmServerlib.h, 124
cache, 15	AmmServerlib/main.c, 66
cacheHashMap, 15	AmmServer_SaveDynamicRequest
clientList, 15	AmmServerlib.h, 124
clientRequestHandlerOverrideContext, 15	AmmServerlib/main.c, 66
files_open, 15	AmmServer_SelfCheck
instanceName, 15	AmmServerlib.h, 124
loaded_cache_items, 15	AmmServerlib/main.c, 68
loaded_cache_items_Kbytes, 15	AmmServer_SetIntSettingValue
pause_server, 15	AmmServerlib.h, 126
prespawn_jobs_finished, 15	AmmServerlib/main.c, 68
prespawn_jobs_started, 15	AmmServer_SetStrSettingValue
prespawn_turn_to_serve, 15	AmmServerlib.h, 126
prespawned_pool, 16	AmmServerlib/main.c, 68
server_running, 16	AmmServer_SignalCountAsBadClientBehaviour
server_thread_id, 16	AmmServerlib.h, 126
serversock, 16	AmmServerlib/main.c, 69
settings, 16	AmmServer_Start
stop_server, 16	AmmServerlib.h, 127
templates_root, 16	AmmServerlib/main.c, 69
threads_pool, 16	AmmServer_StartAdminInstance
webserver_root, 16	AmmServerlib.h, 127
AmmServer_Instance_Settings, 16	AmmServerlib/main.c, 70
BASE64PASSWORD, 16	AmmServer_StartWithArgs
BINDING_PORT, 16	AmmServerlib.h, 128
PASSWORD, 16	AmmServerlib/main.c, 70
USERNAME, 16	AmmServer_Stop
AmmServer_POSTArg	AmmServerlib.h, 128
AmmServerlib.h, 121	AmmServerlib/main.c, 71
AmmServerlib/main.c, 63	AmmServer_StringIsHTMLSafe
AmmServer_PreCacheFile	AmmServerlib.h, 130
AmmServerlib/main.c, 63	AmmServerlib/main.c, 72
AmmServer_RH_Context, 18	AmmServer_Success
callback_cooldown, 18	AmmServerlib.h, 130
callback_every_x_msec, 18	AmmServerlib/main.c, 72
dynamicRequestCallbackFunction, 18	AmmServer_Version
last_callback, 18	AmmServerlib.h, 130
RH_Scenario, 19	AmmServerlib/main.c, 72
requestContext, 18	AmmServer_Warning
resource_name, 18	AmmServerlib.h, 131
web_root_path, 19	AmmServerlib/main.c, 72
AmmServer_ReadFileToMemory	AmmServer_WriteFileFromMemory
AmmServerlib.h, 121	AmmServerlib.h, 131
AmmServerlib/main.c, 63	AmmServerlib/main.c, 74
AmmServer_RegisterTerminationSignal	AmmServerlib.h

	_FILES, 115	POST, 56
	_GET, 115	AmmServer_AddRequestHandler, 56
	_POST, 115	AmmServer AddResourceHandler, 57
	AmmServInfos, 113	AmmServer_CheckIfHeaderBinaryAreTheSame,
	AmmServSettings, 114	57
	AmmServStrSettings, 114	AmmServer_DirectoryExists, 58
	AmmServer_AddRequestHandler, 115	AmmServer_DoNOTCacheResource, 58
	AmmServer_AddResourceHandler, 116	AmmServer_DoNOTCacheResourceHandler, 58
	AmmServer_CheckIfHeaderBinaryAreTheSame,	AmmServer EraseFile, 59
	116	AmmServer Error, 59
	AmmServer_DirectoryExists, 117	AmmServer ExecuteCommandLine, 59
	AmmServer DoNOTCacheResource, 117	AmmServer FILES, 60
	AmmServer_DoNOTCacheResourceHandler, 117	AmmServer_FileExists, 60
	AmmServer_EraseFile, 118	AmmServer_GETArg, 60
	AmmServer_Error, 118	AmmServer_GeneralPrint, 60
	AmmServer_ExecuteCommandLine, 118	AmmServer_GetInfo, 62
	AmmServer_FILES, 119	AmmServer_GetIntSettingValue, 62
	AmmServer_FileExists, 119	AmmServer_GetStrSettingValue, 62
	AmmServer_GETArg, 119	AmmServer_GlobalTerminationHandler, 63
	AmmServer_GetInfo, 120	AmmServer_POSTArg, 63
	AmmServer_GetIntSettingValue, 120	AmmServer_PreCacheFile, 63
	AmmServer_GetStrSettingValue, 120	AmmServer_ReadFileToMemory, 63
	AmmServer_POSTArg, 121	AmmServer_RegisterTerminationSignal, 65
	AmmServer_ReadFileToMemory, 121	AmmServer_RemoveResourceHandler, 65
	AmmServer_RegisterTerminationSignal, 121	AmmServer_ReplaceVarInMemoryFile, 66
	AmmServer_RemoveResourceHandler, 123	AmmServer_Running, 66
	AmmServer_ReplaceVarInMemoryFile, 123	AmmServer_SaveDynamicRequest, 66
	AmmServer_Running, 124	AmmServer_SelfCheck, 68
	AmmServer_SaveDynamicRequest, 124	AmmServer_SetIntSettingValue, 68
	AmmServer_SelfCheck, 124	AmmServer_SetStrSettingValue, 68
	AmmServer_SetIntSettingValue, 126	AmmServer_SignalCountAsBadClientBehaviour,
	AmmServer_SetStrSettingValue, 126	69
	AmmServer_SignalCountAsBadClientBehaviour,	AmmServer_Start, 69
	126	AmmServer_StartAdminInstance, 70
	AmmServer_Start, 127	AmmServer_StartWithArgs, 70
	AmmServer_StartAdminInstance, 127	AmmServer_Stop, 71
	AmmServer_StartWithArgs, 128	AmmServer_StringIsHTMLSafe, 72
	AmmServer_Stop, 128	AmmServer_Success, 72
	AmmServer_StringIsHTMLSafe, 130	AmmServer_Version, 72
	AmmServer_Success, 130	AmmServer_Warning, 72
	AmmServer_Version, 130	AmmServer_WriteFileFromMemory, 74
	AmmServer_Warning, 131	TerminationCallback, 74
	AmmServer_WriteFileFromMemory, 131	AnalyzeHTTPHeader
	MAX_FILE_PATH, 113	http_header_analysis.c, 188
	MAX_QUERY, 113	http_header_analysis.h, 193
	MAX_RESOURCE, 113	AnalyzeHTTPLineRequest
	POPEN_BUFFER_SIZE, 113 RHScenarios, 114	http_header_analysis.c, 188
	TypesOfRequests, 114	AnalyzePOSTLineRequest post_header_analysis.c, 196
۸mn	nServerlib/InputParser/InputParser_C_Tester/main	post_header_analysis.h, 199
AIIII	C	Append2MyURLDBFile
	IntermediateTests, 51	Services/MyURL/main.c, 88
	main, 51	AppendPOSTRequestToHTTPHeader
	max_ret_word, 51	http_header_analysis.c, 189
	ParseString, 52	http_header_analysis.h, 194
Amn	nServerlib/main.c	applicationFiles.h
	_FILES, 55	APPLICATIONFILES_CPL, 247
	_GET, 56	APPLICATIONFILES_DLL, 247
	<del>_</del> ,	= <u> </u>

APPLICATIONFILES_EMPTY, 247	imaging.c, 99
APPLICATIONFILES_END_OF_ITEMS, 248	imaging.h, 100
APPLICATIONFILES EXE, 247	bitBltImageRotated
APPLICATIONFILES_PDF, 247	imaging.c, 99
APPLICATIONFILES_SCR, 247	boundary
APPLICATIONFILES_SWF, 247	HTTPHeader, 25
applicationFiles.c	boundaryLength
scanFor_applicationFiles, 246	HTTPHeader, 25
applicationFiles.h	busy
scanFor_applicationFiles, 248	PreSpawnedThread, 38
AssignStr	
server_configuration.c, 232	CONNECT
server_configuration.h, 243	AmmServerlib.h, 114
audioFiles.h	CACHING_ENABLED
AUDIOFILES_AU, 250	server_configuration.c, 235
AUDIOFILES_EMPTY, 250	server_configuration.h, 245
AUDIOFILES_END_OF_ITEMS, 250	CHANGE_PRIORITY
AUDIOFILES_MID, 250	server_configuration.c, 235
AUDIOFILES_MP3, 250	server_configuration.h, 245
AUDIOFILES_OGG, 250	CHANGE_TO_UID
AUDIOFILES_VOC, 250	server_configuration.c, 235
AUDIOFILES_WAV, 250	server_configuration.h, 245
audioFiles.c	CONTAINERS_MAX
scanFor_audioFiles, 248	InputParser_C.h, 211
audioFiles.h	CR
scanFor_audioFiles, 250	http_header_analysis.c, 188
authorized	CYAN
HTTPHeader, 25	logs.h, 315
	cache
BAD	AmmServer_Instance, 15
AmmServerlib.h, 114	cache_AddDoNOTCacheRuleForResource
BASE64PASSWORD	file_caching.c, 145
AmmServer_Instance_Settings, 16	file_caching.h, 153
BINDING_PORT	cache_AddFile
AmmServer_Instance_Settings, 16	file_caching.c, 145
BLACK	file_caching.h, 154
logs.h, 315	cache_AddMemoryBlock
BLUE	file_caching.c, 146
logs.h, 315	file_caching.h, 154
BOLDBLACK	cache_ChangeRequestIfTemplateRequested
logs.h, 315	file_caching.c, 146
BOLDBLUE	file_caching.h, 155
logs.h, 315	cache_CreateResource
BOLDCYAN	file_caching.c, 147
logs.h, 315	cache_Destroy
BOLDGREEN	file_caching.c, 147
logs.h, 315	file_caching.h, 155
BOLDMAGENTA	cache_DestroyResource
logs.h, 315	file_caching.c, 147
BOLDRED	cache_FindResource
logs.h, 315	file_caching.c, 147
BOLDWHITE	file_caching.h, 156
logs.h, 315	cache_GetHashOfResource
BOLDYELLOW	file_caching.c, 148
logs.h, 315	file_caching.h, 156
base_image	cache_GetResource
ScriptRunner/main.cpp, 326	file_caching.c, 148
Services/ScriptRunner/main.cpp, 332	file_caching.h, 156
bitBltImage	cache_Initialize

file_caching.c, 149	hashMap, 23
file_caching.h, 157	client
cache_LoadResourceFromDisk	PassToHTTPThread, 35
file_caching.c, 150	PreSpawnedThread, 38
cache RemoveContextAndResource	client_list.c
file_caching.c, 150	clientList_GetClientId, 134
file_caching.h, 158	clientList_close, 132
cache_RemoveResource file caching.c, 150	clientList_initialize, 134 clientList_isClientAllowedToUseResource, 134
cache_ResourceExists	clientList_isClientBanned, 134
file_caching.c, 151	clientList_isolientStoppedUsingResource, 135
file_caching.h, 158	client_list.h
cache_item, 19	clientID, 136
compressedContent, 20	clientList_GetClientId, 137
compressedContentSize, 20	clientList_close, 136
content, 20	clientList_initialize, 137
content/Size, 20	clientList_isClientAllowedToUseResource, 137
contentTypeID, 20	clientList_isClientBanned, 137
doNOTCacheRule, 20	clientList_signalClientStoppedUsingResource, 138
dynamicRequest, 20	clientID
dynamicRequestCallbackFunction, 20	AmmServer_DynamicRequest, 13
modification, 20	client list.h, 136
cacheHashMap	clientList
AmmServer Instance, 15	AmmServer_Instance, 15
callClientRequestHandler	clientList_GetClientId
dynamic_requests.c, 139	client_list.c, 134
dynamic_requests.h, 142	client list.h, 137
callback_cooldown	clientList close
AmmServer_RH_Context, 18	client_list.c, 132
callback_every_x_msec	client_list.h, 136
AmmServer_RH_Context, 18	clientList initialize
captcha_url	client_list.c, 134
Services/MyURL/main.c, 94	client list.h, 137
captchaStrings	clientList_isClientAllowedToUseResource
AmmCaptcha/main.c, 50	client list.c, 134
chatbox	client list.h, 137
ScriptRunner/main.cpp, 326	clientList_isClientBanned
Services/AmmarServer/main.c, 78	client_list.c, 134
Services/ScriptRunner/main.cpp, 332	client_list.h, 137
CheckDelimeterNumOk	clientList_signalClientStoppedUsingResource
InputParser_C.c, 203	client_list.c, 135
CheckHTTPHeaderCategory	client_list.h, 138
http_tools.c, 293	clientListContext, 20
http_tools.h, 303	userList, 21
CheckHTTPHeaderCategoryAllCaps	clientListID
http_tools.c, 293	HTTPTransaction, 27
http_tools.h, 303	clientRequestHandlerOverrideContext
CheckIPCOk	AmmServer_Instance, 15
InputParser_C.c, 203	clientSock
CheckWordNumOk	HTTPTransaction, 27
InputParser_C.c, 203	clientlen
InputParser_C.h, 211	PassToHTTPThread, 35
checksum	PreSpawnedThread, 38
guard_byte, 23	clientsock
childFinishedWithParentMessage	PassToHTTPThread, 35
threadInitHelper.c, 285	PreSpawnedThread, 38
threadinit Helper.h, 287	close_dynamic_content
clearItemCallbackFunction	helloworld.c, 42
SISALITOTHISAIDASIN AHISTOTI	Honoworlding, 12

ScriptRunner/main.cpp, 322	create url
·	<del>-</del>
Services/AmmarServer/main.c, 76	Services/MyURL/main.c, 94
Services/GeoPosShare/main.c, 80	CreateCompressedVersionofCachedResource
Services/MyLoader/main.c, 83	file_compression.c, 160
Services/MyURL/main.c, 88	CreateCompressedVersionofDynamicContent
Services/ScriptRunner/main.cpp, 328	file_compression.c, 160
Services/SimpleTemplate/main.c, 96	file_compression.h, 163
cmpHashTableItems	CreateCompressedVersionofStaticContent
hashmap.c, 166	file_compression.c, 160
compressedContent	file_compression.h, 163
AmmServer_DynamicRequest, 13	CreateCompressedVersionofStaticContentPreloading
cache_item, 20	file_compression.c, 160
compressedContentSize	file_compression.h, 164
AmmServer_DynamicRequest, 13	createImage
cache_item, 20	imaging.c, 99
container_end	imaging.h, 100
InputParserC, 33	cur_container_count
container_start	InputParserC, 33
InputParserC, 33	cur_delimeter_count
content	InputParserC, 33
AmmServer_DynamicRequest, 13	curNumberOfEntries
cache item, 20	hashMap, 23
contentDisposition	
HTTPHeader, 25	DELETE
contentDispositionLength	AmmServerlib.h, 114
HTTPHeader, 26	DIFFERENT_PAGE_FOR_EACH_CLIENT
	AmmServerlib.h, 114
ContentLength	DELIM_MAX_MAX
HTTPHeader, 26	InputParser_C.h, 211
contentSize	DISPLAY_DEBUG_INFO
AmmServer_DynamicRequest, 13	imaging.c, 99
cache_item, 20	day
contentType	timestamp, 39
http_tools.h, 303	days
HTTPHeader, 26	time_provider.c, 317
contentTypeEnumerator	db_addIDLock
http_tools.h, 303	Services/MyURL/main.c, 94
contentTypeID	db_file
cache_item, 20	Services/MyURL/main.c, 94
contentTypeLength	db_fileLock
HTTPHeader, 26	Services/MyURL/main.c, 94
contents	default_failed
fastStringParser, 21	Services/MyURL/main.c, 94
convertExternalIDToInternal	default_server
AmmCaptcha/main.c, 49	ScriptRunner/main.cpp, 326
convertTo_ENUM_ID	Services/AmmarServer/main.c, 78
fastStringParser.c, 334	Services/GeoPosShare/main.c, 81
convertToUpperCase	Services/MyLoader/main.c, 85
http_tools.c, 294	Services/ScriptRunner/main.cpp, 332
cookie	Services/SimpleTemplate/main.c, 97
HTTPHeader, 26	DefaultDelimeterSetup
cookieLength	InputParser, 29
HTTPHeader, 26	delimeters
coolPHPWave	InputParserC, 33
img_warp.c, 101	depth
img_warp.h, 103	Image, 28
copylmage	destroyImage
imaging.c, 99	imaging.c, 100
imaging.h, 100	imaging.h, 100

difference	server_configuration.c, 232
time_snap, 38	server_configuration.h, 243
directory_lists.c	empty_buffer
GenerateDirectoryPage, 289	jpglnput.c, 105
path_cat, 290	encodeToBase64
starting, 289	http_tools.c, 294
tag_after_image, 289	http_tools.h, 304
tag_pre_image, 289	end timer
directory_lists.h	time_provider.c, 316
GenerateDirectoryPage, 291	time_provider.h, 319
DirectoryExistsAmmServ	entries
http_tools.c, 294	hashMap, 23
http_tools.h, 304	entryAllocationStep
doNOTCacheRule	hashMap, 23
cache_item, 20	EraseFile
doc/DoxygenMainpage.h, 41	Services/ScriptRunner/main.cpp, 328
doc/helloworld.c, 41	error
dynamic_requests.c	logs.c, 312
callClientRequestHandler, 139	logs.h, 315
dynamicRequest_ContentAvailiable, 139	error_url
dynamicRequest_serveContent, 139	Services/MyURL/main.c, 94
saveDynamicRequest, 140	ErrorLog
dynamic_requests.h	server_configuration.c, 235
callClientRequestHandler, 142	server_configuration.h, 245
dynamicRequest_ContentAvailiable, 142	ErrorLogAppend
dynamicRequest_serveContent, 142	logs.c, 312
saveDynamicRequest, 143	logs.h, 315
dynamicRequest	ErrorLogEnable
cache_item, 20	server_configuration.c, 235
dynamicRequest_ContentAvailiable	server_configuration.h, 245
dynamic_requests.c, 139	execute
dynamic_requests.h, 142	ScriptRunner/main.cpp, 322
dynamicRequest_serveContent	Services/ScriptRunner/main.cpp, 328
dynamic_requests.c, 139	executeScript
dynamic_requests.h, 142	Services/AmmarServer/main.c, 78
dynamicRequestCallbackFunction	executeScriptFunction
AmmServer_RH_Context, 18	Services/AmmarServer/main.c, 76
cache_item, 20	executeScriptRC
EXECUTABLE	Services/AmmarServer/main.c, 79
http_tools.h, 303	export_C_Scanner
ENABLE_ADMIN_PAGE	fastStringParser.c, 334
ScriptRunner/main.cpp, 322	fastStringParser.h, 337
Services/AmmarServer/main.c, 76	FIRSTLINES CONNECT
Services/GeoPosShare/main.c, 80	firstLines.h, 252
Services/ScriptRunner/main.cpp, 328	FIRSTLINES_DELETE
ENABLE CHAT BOX	firstLines.h, 252
ScriptRunner/main.cpp, 322	FIRSTLINES EMPTY
Services/AmmarServer/main.c, 76	firstLines.h, 252
Services/ScriptRunner/main.cpp, 328	FIRSTLINES_END_OF_ITEMS
ENABLE_COMPRESSION	firstLines.h, 252
server_configuration.h, 240	FIRSTLINES_GET
ENABLE_POST	firstLines.h, 252
server_configuration.h, 240	FIRSTLINES_HEAD
eTag	firstLines.h, 252
HTTPHeader, 26	FIRSTLINES_OPTIONS
eTagLength	firstLines.h, 252
HTTPHeader, 26	FIRSTLINES_PATCH
EmmitPossibleConfigurationWarnings	firstLines.h, 252

FIRSTLINES_POST	cache_DestroyResource, 147
firstLines.h, 252	cache_FindResource, 147
FIRSTLINES_PUT	cache_GetHashOfResource, 148
firstLines.h, 252	cache_GetResource, 148
FIRSTLINES_TRACE	cache_Initialize, 149
firstLines.h, 252	cache_LoadResourceFromDisk, 150
FOLDER	cache_RemoveContextAndResource, 150
http_tools.h, 303	cache_RemoveResource, 150
fastJPGHeaderCheck	cache_ResourceExists, 151
jpgInput.c, 105	freeMallocIfNeeded, 151
fastSTringParser_createRulesFromFile	file_caching.h
fastStringParser.c, 335	cache_AddDoNOTCacheRuleForResource, 153
fastStringParser.h, 338	cache_AddFile, 154
fastStringParser, 21	cache_AddMemoryBlock, 154
contents, 21	cache_ChangeRequestIfTemplateRequested, 155
functionName, 21	cache_Destroy, 155
longestStringLength, 22	cache_FindResource, 156
MAXstringsLoaded, 22	cache_GetHashOfResource, 156
shortestStringLength, 22	cache_GetResource, 156
stringsLoaded, 22	cache_Initialize, 157
fastStringParser.c	cache_RemoveContextAndResource, 158
ACTIVATED_LEVELS, 333	cache_ResourceExists, 158
acceptedChars, 336	freeMallocIfNeeded, 159
addLevelSpaces, 334	file_compression.c
convertTo_ENUM_ID, 334	CreateCompressedVersionofCachedResource,
export_C_Scanner, 334	160
fastSTringParser_createRulesFromFile, 335	CreateCompressedVersionofDynamicContent, 160
fastStringParser_addString, 334	CreateCompressedVersionofStaticContent, 160
fastStringParser_close, 334	CreateCompressedVersionofStaticContentPreloading
fastStringParser_countStringsForNextChar, 334	160
fastStringParser_hasStringsWithNConsecutive-	file_compression.h
Chars, 335	CreateCompressedVersionofDynamicContent, 163
fastStringParser_initialize, 335	CreateCompressedVersionofStaticContent, 163
fspHTTPHeader, 336	CreateCompressedVersionofStaticContentPreloading
MAXIMUM_LEVELS, 333	164
printAllEnumeratorItems, 335	file_server.c
printIfAllPossibleStrings, 335	files_open, 221
recursiveTraverser, 336	SendErrorFile, 218
fastStringParser.h	SendFile, 218
export_C_Scanner, 337	SendMemoryBlockAsFile, 219
fastSTringParser_createRulesFromFile, 338	SendPart, 220
fastStringParser_addString	TransmitFileToSocket, 220
fastStringParser.c, 334	TransmitFileToSocketInternal, 220
fastStringParser_close	file_server.h
fastStringParser.c, 334	SendErrorFile, 223
fastStringParser_countStringsForNextChar	SendFile, 223
fastStringParser.c, 334	SendMemoryBlockAsFile, 224
fastStringParser_hasStringsWithNConsecutiveChars	FileExistsAmmServ
fastStringParser.c, 335	http_tools.c, 294
fastStringParser_initialize	http_tools.h, 304
fastStringParser.c, 335	FileExistsTest
file_caching.c	Services/ScriptRunner/main.cpp, 329
cache_AddDoNOTCacheRuleForResource, 145	FilenameStripperOk
cache_AddFile, 145	http_tools.c, 295
cache_AddMemoryBlock, 146	http_tools.h, 304
cache_ChangeRequestIfTemplateRequested, 146	files_open
cache_CreateResource, 147	AmmServer_Instance, 15
cache_Destroy, 147	file_server.c, 221

Find_longURL	functionName
Services/MyURL/main.c, 88	fastStringParser, 21
Find_longURLSerial	lasioningi arser, 21
Services/MyURL/main.c, 89	GET
FindAProperThreadID	AmmServerlib.h, 114
freshThreads.c, 267	GET override
FindIndexFile	ScriptRunner/main.cpp, 326
	Services/AmmarServer/main.c, 79
http_tools.c, 295	Services/GeoPosShare/main.c, 82
http_tools.h, 306	Services/MyLoader/main.c, 85
findOutClientIDOfPeer	Services/ScriptRunner/main.cpp, 332
http_tools.c, 295	Services/SimpleTemplate/main.c, 97
http_tools.h, 306	GET_request
firstLines.h	AmmServer_DynamicRequest, 13
FIRSTLINES_CONNECT, 252	GET_request_length
FIRSTLINES_DELETE, 252	AmmServer_DynamicRequest, 14
FIRSTLINES_EMPTY, 252	GETquery
FIRSTLINES_END_OF_ITEMS, 252	HTTPHeader, 26
FIRSTLINES_GET, 252	GREEN
FIRSTLINES_HEAD, 252	logs.h, 315
FIRSTLINES_OPTIONS, 252	GenerateDirectoryPage
FIRSTLINES_PATCH, 252	directory_lists.c, 289
FIRSTLINES_POST, 252	directory_lists.h, 291
FIRSTLINES_PUT, 252	Get_longURL
FIRSTLINES_TRACE, 252	Services/MyURL/main.c, 89
firstLines.c	getBackCommandLine
scanFor_firstLines, 251	ScriptRunner/main.cpp, 322
firstLines.h	Services/ScriptRunner/main.cpp, 329
scanFor_firstLines, 252	GetContentType
fontRAW	http_tools.c, 296
AmmCaptcha/main.c, 50	http_tools.h, 307
fontX	GetContentTypeForExtension
AmmCaptcha/main.c, 50	http_tools.c, 296
fontY	GetDateString
AmmCaptcha/main.c, 50	http_tools.h, 307
form	time_provider.c, 317
ScriptRunner/main.cpp, 326	time provider.h, 319
Services/AmmarServer/main.c, 79	GetDelimeter
Services/ScriptRunner/main.cpp, 332	InputParser, 30
FreeHTTPHeader	GetExtensionImage
http_header_analysis.c, 189	http_tools.c, 297
http_header_analysis.h, 194	http_tools.h, 307
freeMallocIfNeeded	GetExtentionType
file caching.c, 151	http_tools.c, 297
file_caching.h, 159	http_tools.h, 308
freeString	GetIntFromHTTPHeaderFieldPayload
http_tools.c, 296	http_tools.c, 298
http_tools.h, 306	http_tools.h, 308
freshThreads.c	GetLowercaseWord
FindAProperThreadID, 267	InputParser, 30
SpawnThreadToServeNewClient, 267	GetNewStringFromHTTPHeaderFieldPayload
freshThreads.h	http_tools.c, 298
SpawnThreadToServeNewClient, 270	http_tools.h, 308
fspHTTPHeader	GetTickCountAmmServ
fastStringParser.c, 336	
_	time_provider.c, 317
fspString, 22	time_provider.h, 319
str, 22	GetUpcaseWord
strIDFriendly, 22	InputParser, 30
strLength, 22	GetWord

InputParser, 30	contentTypeLength, 26
GetWordChar	cookie, 26
InputParser, 31	cookieLength, 26
GetWordInt	eTag, 26
InputParser, 31	eTagLength, 26
GetWordLength	GETquery, 26
InputParser, 31	headerRAW, 26
goto_url	headerRAWSize, 26
Services/MyURL/main.c, 94	host, 26
gps	hostLength, 26
Services/AmmarServer/main.c, 79	keepalive, 26
Services/GeoPosShare/main.c, 82	POSTrequest, 26
guard_byte, 22	POSTrequestSize, 26
checksum, 23	range_end, 26
guardbyte1	range_start, 26
InputParserC, 34	<del>-</del> -
guardbyte2	referer, 26
InputParserC, 34	refererLength, 26
guardbyte3	requestType, 26
InputParserC, 34	resource, 26
guardbyte4	supports_compression, 26
InputParserC, 34	userAgent, 26
inputi discre, or	userAgentLength, 26
HEAD	verified_local_resource, 26
AmmServerlib.h, 114	HTTPHeaderComplete
HTTPHEADER_ACCEPT_ENCODING	http_header_analysis.c, 190
httpHeader.h, 254	http_header_analysis.h, 195
HTTPHEADER AUTHORIZATION	HTTPHeaderlsPOST
httpHeader.h, 254	http_header_analysis.c, 190
HTTPHEADER CONNECTION	http_header_analysis.h, 195
httpHeader.h, 255	HTTPServerIsRunning
HTTPHEADER COOKIE	threadedServer.c, 276
httpHeader.h, 254	threadedServer.h, 282
HTTPHEADER EMPTY	HTTPTransaction, 27
httpHeader.h, 254	clientListID, 27
HTTPHEADER_END_OF_ITEMS	clientSock, 27
httpHeader.h, 255	incomingHeader, 27
HTTPHEADER_HOST	instance, 28
httpHeader.h, 255	outgoingBody, 28
HTTPHEADER_IF_MODIFIED_SINCE	outgoingBodySize, 28
httpHeader.h, 255	prespawnedThreadFlag, 28
HTTPHEADER_IF_NONE_MATCH	resourceCacheID, 28
httpHeader.h, 255	threadID, 28
HTTPHEADER RANGE	hashFunction
httpHeader.h, 255	hashmap.c, 166
HTTPHEADER_REFERER	hashmap.h, 178
httpHeader.h, 255	hashMap, 23
HTTPHEADER REFERRER	clearItemCallbackFunction, 23
httpHeader.h, 255	curNumberOfEntries, 23
HTTPHEADER_USER_AGENT	entries, 23
	entryAllocationStep, 23
httpHeader.h, 255	hm_addLock, 24
HTTPHeader, 25	
authorized, 25	hm_fileLock, 24
boundary, 25	maxNumberOfEntries, 24
boundaryLength, 25	hashMap_Add
contentDisposition, 25	hashmap.c, 166
contentDispositionLength, 26	hashmap.h, 178
ContentLength, 26	hashMap_AddULong
contentType, 26	hashmap.c, 167

hashmap.h, 178	payloadLength, 24
hashMap_Clear	hashURL
hashmap.c, 167	Services/MyURL/main.c, 89
hashmap.h, 179	hashmap.c
hashMap_ContainsKey	cmpHashTableItems, 166
hashmap.c, 168	hashFunction, 166
hashmap.h, 179	hashMap_Add, 166
hashMap_ContainsValue	hashMap_AddULong, 167
hashmap.c, 168	hashMap_Clear, 167
hashmap.h, 180	hashMap_ContainsKey, 168
hashMap_Create	hashMap_ContainsValue, 168
hashmap.c, 169	hashMap_Create, 169
hashmap.h, 180	hashMap_Destroy, 169
hashMap_Destroy	hashMap_FindIndex, 170
hashmap.c, 169	hashMap_GetCurrentNumberOfEntries, 170
hashmap.h, 181	hashMap_GetHashAtIndex, 171
hashMap_FindIndex	hashMap_GetKeyAtIndex, 171
hashmap.c, 170	hashMap_GetMaxNumberOfEntries, 172
hashmap.h, 181	hashMap_GetPayload, 172
hashMap_GetCurrentNumberOfEntries	hashMap_GetULongPayload, 173
hashmap.c, 170	hashMap_Grow, 173
hashmap.h, 182	hashMap_lsOK, 173
hashMap_GetHashAtIndex	hashMap_IsSorted, 173
hashmap.c, 171	hashMap_LoadToFile, 174
hashmap.h, 182	hashMap_SaveToFile, 174
hashMap_GetKeyAtIndex	hashMap_Sort, 175
hashmap.c, 171	hashmap_SwapRecords, 175
hashmap.h, 183	hashmap.h
hashMap_GetMaxNumberOfEntries	hashFunction, 178
hashmap.c, 172	hashMap_Add, 178
hashmap.h, 183	hashMap_AddULong, 178
hashMap_GetPayload	hashMap_Clear, 179
hashmap.c, 172	hashMap_ContainsKey, 179
hashmap.h, 184	hashMap_ContainsValue, 180
hashMap_GetULongPayload	hashMap_Create, 180
hashmap.c, 173	hashMap_Destroy, 181
hashmap.h, 184	hashMap_FindIndex, 181
hashMap_Grow	hashMap_GetCurrentNumberOfEntries, 182
hashmap.c, 173	hashMap_GetHashAtIndex, 182
hashMap_IsOK	hashMap_GetKeyAtIndex, 183
hashmap.c, 173	hashMap_GetMaxNumberOfEntries, 183
hashMap_IsSorted	hashMap_GetPayload, 184
hashmap.c, 173	hashMap_GetULongPayload, 184
hashMap_LoadToFile	hashMap_LoadToFile, 185
hashmap.c, 174	hashMap_SaveToFile, 185
hashmap.h, 185	hashMap_Sort, 186
hashMap_SaveToFile	hashmap_SwapRecords, 186
hashmap.c, 174	hashmap_SwapRecords
hashmap.h, 185	hashmap.c, 175 hashmap.h, 186
hashMap_Sort	headerRAW
hashmap.c, 175	
hashmap.h, 186 hashMapEntry, 24	HTTPHeader, 26 headerRAWSize
	HTTPHeader, 26
hits, 24 key, 24	headerResponse
keyHash, 24	AmmServer_DynamicRequest, 14
keyLength, 24	— ·
payload, 24	height Image, 28
payluau, <del>24</del>	illaye, 20

helloworld	ProcessFirstHTTPLine, 190
helloworld.c, 43	ProcessRangeHTTPLine, 191
helloworld.c	ReceiveHTTPHeader, 191
close_dynamic_content, 42	http_header_analysis.h
helloworld, 43	AnalyzeHTTPHeader, 193
helloworld_times_shown, 43	AppendPOSTRequestToHTTPHeader, 194
init_dynamic_content, 42	FreeHTTPHeader, 194
main, 42	HTTPHeaderComplete, 195
prepare_helloworld_content_callback, 42	HTTPHeaderIsPOST, 195
templates_root, 43	ReceiveHTTPHeader, 195
webserver_root, 43	http_tools.c
helloworld times shown	CheckHTTPHeaderCategory, 293
helloworld.c, 43	CheckHTTPHeaderCategoryAllCaps, 293
hits	convertToUpperCase, 294
	DirectoryExistsAmmServ, 294
hashMapEntry, 24	
hm_addLock	encodeToBase64, 294
hashMap, 24	FileExistsAmmServ, 294
hm_fileLock	FilenameStripperOk, 295
hashMap, 24	FindIndexFile, 295
host	findOutClientIDOfPeer, 295
HTTPHeader, 26	freeString, 296
hostLength	GetContentType, 296
HTTPHeader, 26	GetContentTypeForExtension, 296
hour	GetExtensionImage, 297
timestamp, 39	GetExtentionType, 297
http_tools.h	GetIntFromHTTPHeaderFieldPayload, 298
AUDIO, 303	GetNewStringFromHTTPHeaderFieldPayload, 298
EXECUTABLE, 303	ReducePathSlashes_Inplace, 298
FOLDER, 303	RequestHTTPWebPage, 299
IMAGE, 303	seek_blank_char, 299
NO_FILETYPE, 303	seek_non_blank_char, 299
RESERVED_CTE_VALUE, 303	ServerThreads_DropRootUID, 299
TEXT, 303	setSocketTimeouts, 299
VIDEO, 303	strToUpcase, 301
httpHeader.h	StripGETRequestQueryAndFragment, 300
HTTPHEADER_ACCEPT_ENCODING, 254	StripHTMLCharacters_Inplace, 300
HTTPHEADER_AUTHORIZATION, 254	StripVariableFromGETorPOSTString, 300
HTTPHEADER_CONNECTION, 255	stristr, 300
HTTPHEADER_COOKIE, 254	stristr2Caps, 301
HTTPHEADER_EMPTY, 254	trim_last_empty_chars, 301
HTTPHEADER_END_OF_ITEMS, 255	http_tools.h
HTTPHEADER_HOST, 255	CheckHTTPHeaderCategory, 303
HTTPHEADER_IF_MODIFIED_SINCE, 255	CheckHTTPHeaderCategoryAllCaps, 303
HTTPHEADER_IF_NONE_MATCH, 255	contentType, 303
HTTPHEADER_RANGE, 255	contentTypeEnumerator, 303
HTTPHEADER_REFERER, 255	DirectoryExistsAmmServ, 304
HTTPHEADER_REFERRER, 255	encodeToBase64, 304
HTTPHEADER_USER_AGENT, 255	FileExistsAmmServ, 304
http_header_analysis.c	FilenameStripperOk, 304
AnalyzeHTTPHeader, 188	FindIndexFile, 306
AnalyzeHTTPLineRequest, 188	findOutClientIDOfPeer, 306
AppendPOSTRequestToHTTPHeader, 189	freeString, 306
CR, 188	GetContentType, 307
FreeHTTPHeader, 189	GetDateString, 307
HTTPHeaderComplete, 190	GetExtensionImage, 307
HTTPHeaderIsPOST, 190	GetExtentionType, 308
LF, 188	GetIntFromHTTPHeaderFieldPayload, 308
ProcessAuthorizationHTTPLine, 190	GetNewStringFromHTTPHeaderFieldPayload, 308

ReducePathSlashes_Inplace, 309	width, 28
RequestHTTPWebPage, 309	imageFiles.h
seek_blank_char, 309	IMAGEFILES_BMP, 257
seek_non_blank_char, 310	IMAGEFILES_DIB, 257
ServerThreads_DropRootUID, 310	IMAGEFILES_EMPTY, 257
setSocketTimeouts, 310	IMAGEFILES_END_OF_ITEMS, 257
strToUpcase, 311	IMAGEFILES_GIF, 257
StripGETRequestQueryAndFragment, 310	IMAGEFILES ICO, 257
StripHTMLCharacters_Inplace, 310	IMAGEFILES J2C, 257
StripVariableFromGETorPOSTString, 311	IMAGEFILES JPEG, 257
trim_last_empty_chars, 311	IMAGEFILES JPG, 257
httpHeader.c	IMAGEFILES PNG, 257
scanFor_httpHeader, 253	IMAGEFILES PNM, 257
httpHeader.h	IMAGEFILES PPM, 257
scanFor_httpHeader, 255	IMAGEFILES_FFM, 257
554.11 51_11tp1154351, 255	<del>-</del> · · ·
IMAGE	IMAGEFILES_RLE, 257
http tools.h, 303	IMAGEFILES_SVG, 257
IMAGEFILES BMP	IMAGEFILES_TIFF, 257
imageFiles.h, 257	IMAGEFILES_WEBP, 257
IMAGEFILES DIB	imageFiles.c
imageFiles.h, 257	scanFor_imageFiles, 256
IMAGEFILES EMPTY	imageFiles.h
imageFiles.h, 257	scanFor_imageFiles, 257
IMAGEFILES_END_OF_ITEMS	imageSize
	Image, 28
imageFiles.h, 257	imaging.c
IMAGEFILES_GIF	bitBltImage, 99
imageFiles.h, 257	bitBltImageRotated, 99
IMAGEFILES_ICO	copylmage, 99
imageFiles.h, 257	createImage, 99
IMAGEFILES_J2C	DISPLAY_DEBUG_INFO, 99
imageFiles.h, 257	destroyImage, 100
IMAGEFILES_JPEG	
imageFiles.h, 257	PPMREADBUFLEN, 99
IMAGEFILES_JPG	ReadPPM, 100
imageFiles.h, 257	WritePPM, 100
IMAGEFILES_PNG	imaging.h
imageFiles.h, 257	bitBltImage, 100
IMAGEFILES_PNM	copylmage, 100
imageFiles.h, 257	createImage, 100
IMAGEFILES_PPM	destroyImage, 100
imageFiles.h, 257	ReadPPM, 100
IMAGEFILES RAW	WritePPM, 100
imageFiles.h, 257	img_warp.c
IMAGEFILES RLE	ABS, 101
imageFiles.h, 257	ABSDIFF, 101
IMAGEFILES SVG	coolPHPWave, 101
imageFiles.h, 257	warpImage, 102
IMAGEFILES TIFF	img_warp.h
imageFiles.h, 257	coolPHPWave, 103
IMAGEFILES_WEBP	warpImage, 103
	incomingHeader
imageFiles.h, 257	HTTPTransaction, 27
i_adapt	
PassToPreSpawnedThread, 36	indexPage
Image, 28	ScriptRunner/main.cpp, 326
depth, 28	Services/MyURL/main.c, 94
height, 28	Services/ScriptRunner/main.cpp, 332
imageSize, 28	indexPageLength
pixels, 28	Services/MyURL/main.c, 94

indexPagePath	InputParser_WordCompare, 208
Services/MyURL/main.c, 94 init buffer	InputParser_WordCompareAuto, 208 InputParser WordCompareNoCase, 209
<del>-</del>	· – ·
jpgInput.c, 105 init dynamic content	InputParser_WordCompareNoCaseAuto, 209 InputParserC_Version, 209
helloworld.c, 42	Str2Int_internal, 209
ScriptRunner/main.cpp, 323	
Services/AmmarServer/main.c, 76	warningsAboutIncorrectlyAllocatedStackIssued, 209
Services/Animal Server/main.c, 76 Services/GeoPosShare/main.c, 80	InputParser C.h
Services/MyLoader/main.c, 83	CONTAINERS MAX, 211
	CheckWordNumOk, 211
Services/MyURL/main.c, 90 Services/ScriptRunner/main.cpp, 329	
• • • • • • • • • • • • • • • • • • • •	DELIM_MAX_MAX, 211
Services/SimpleTemplate/main.c, 96	InputParser_ClearNonCharacters, 211
InputParser, 28	InputParser_Create, 212
~InputParser, 29	InputParser_DefaultDelimeters, 212
DefaultDelimeterSetup, 29	InputParser_Destroy, 212
GetDelimeter, 30	InputParser_GetDelimeter, 212
GetLowercaseWord, 30	InputParser_GetLowercaseWord, 212
GetUpcaseWord, 30	InputParser_GetUpcaseWord, 213
GetWord, 30	InputParser_GetWord, 213
GetWordChar, 31	InputParser_GetWordChar, 213
GetWordInt, 31	InputParser_GetWordFloat, 213
GetWordLength, 31	InputParser_GetWordInt, 214
InputParser, 29	InputParser_GetWordLength, 214
InputParser, 29	InputParser_SelfCheck, 214
SeperateWords, 31	InputParser_SeperateWords, 214
SeperateWordsCC, 32	InputParser_SeperateWordsCC, 215
SeperateWordsUC, 32	InputParser_SeperateWordsUC, 215
SetDelimeter, 32	InputParser_SetDelimeter, 215
Version, 32	InputParser_TrimCharacters, 215
InputParser.cpp	InputParser_TrimCharactersEnd, 216
ver, 201	InputParser_TrimCharactersStart, 216
Version, 201	InputParser_WordCompare, 216
InputParser_C.c	InputParser_WordCompareAuto, 216
_ipc_ver, 209	InputParser_WordCompareNoCase, 216
CheckDelimeterNumOk, 203	InputParser_WordCompareNoCaseAuto, 216
CheckIPCOk, 203	InputParserC_Version, 217
CheckWordNumOk, 203	MAX_COMPLICITY, 211
InputParser_ClearNonCharacters, 204	MAX_MEMORY, 211
InputParser_Create, 204	MAX_STRING, 211
InputParser_DefaultDelimeters, 204	InputParser_ClearNonCharacters
InputParser_Destroy, 204	InputParser_C.c, 204
InputParser_GetDelimeter, 204	InputParser_C.h, 211
InputParser_GetLowercaseWord, 205	InputParser_Create
InputParser_GetUpcaseWord, 205	InputParser_C.c, 204
InputParser_GetWord, 205	InputParser_C.h, 212
InputParser_GetWordChar, 205	InputParser_DefaultDelimeters
InputParser_GetWordFloat, 206	InputParser_C.c, 204
InputParser_GetWordInt, 206	InputParser_C.h, 212
InputParser_GetWordLength, 206	InputParser_Destroy
InputParser_SelfCheck, 206	InputParser_C.c, 204
InputParser_SeperateWords, 207	InputParser_C.h, 212
InputParser_SeperateWordsCC, 207	InputParser_GetDelimeter
InputParser_SeperateWordsUC, 207	InputParser_C.c, 204
InputParser_SetDelimeter, 207	InputParser_C.h, 212
InputParser_TrimCharacters, 208	InputParser_GetLowercaseWord
InputParser_TrimCharactersEnd, 208	InputParser_C.c, 205
InputParser_TrimCharactersStart, 208	InputParser_C.h, 212

InputParser_GetUpcaseWord	cur_delimeter_count, 33
InputParser_C.c, 205	delimeters, 33
InputParser_C.h, 213	guardbyte1, 34
InputParser_GetWord	guardbyte2, 34
InputParser_C.c, 205	guardbyte3, 34
InputParser_C.h, 213	guardbyte4, 34
InputParser_GetWordChar	local_allocation, 34
InputParser_C.c, 205	max_container_count, 34
InputParser_C.h, 213	max_delimeter_count, 34
InputParser_GetWordFloat	str, 34
InputParser C.c, 206	str_length, 34
InputParser C.h, 213	tokenlist, 34
InputParser_GetWordInt	tokens_count, 34
InputParser_C.c, 206	tokens_max, 34
InputParser_C.h, 214	InputParserC_Version
InputParser_GetWordLength	InputParser_C.c, 209
InputParser_C.c, 206	InputParser_C.h, 217
InputParser_C.h, 214	instance
InputParser_SelfCheck	HTTPTransaction, 28
InputParser_C.c, 206	PassToHTTPThread, 35
InputParser_C.h, 214	PassToPreSpawnedThread, 36
InputParser_SeperateWords	PreSpawnedThread, 38
InputParser_C.c, 207	instance_CountFreeOP
InputParser C.h, 214	server_configuration.c, 232
InputParser_SeperateWordsCC	server_configuration.h, 243
InputParser_C.c, 207	instance_CountNewMallocOP
InputParser_C.h, 215	server_configuration.c, 234
InputParser_SeperateWordsUC	server_configuration.h, 243
InputParser_C.c, 207	instance_WeCanCommitMoreMemory
InputParser_C.h, 215	server_configuration.c, 234
InputParser_SetDelimeter	server_configuration.h, 243
InputParser_C.c, 207	instanceName
InputParser_C.h, 215	AmmServer_Instance, 15
InputParser_TrimCharacters	IntermediateTests
InputParser_C.c, 208	AmmServerlib/InputParser/InputParser_C
InputParser_C.h, 215	Tester/main.c, 51
InputParser_TrimCharactersEnd	is_an_unsafe_str
InputParser_C.c, 208	Services/MyURL/main.c, 90
InputParser_C.h, 216	isURLDBSorted
InputParser_TrimCharactersStart	Services/MyURL/main.c, 90
InputParser_C.c, 208	and the same of th
InputParser_C.h, 216	joystickExecute
InputParser_WordCompare	ScriptRunner/main.cpp, 323
InputParser_C.c, 208	Services/ScriptRunner/main.cpp, 329
InputParser_C.h, 216	jpegtest
InputParser_WordCompareAuto	jpgInput.c, 105
InputParser_C.c, 208	jpgInput.c
InputParser C.h, 216	empty_buffer, 105
InputParser_WordCompareNoCase	fastJPGHeaderCheck, 105
	init_buffer, 105
InputParser_C.c, 209	jpegtest, 105
InputParser_C.h, 216	ReadJPEG, 105
InputParser_WordCompareNoCaseAuto	term_buffer, 105
InputParser_C.c, 209	Write IREClaternal, 106
InputParser_C.h, 216	Write IRECMomery 106
InputParserC, 33	WriteJPEGMemory, 106
container_end, 33	jpgInput.h
container_start, 33	ReadJPEG, 108
cur_container_count, 33	USE_JPG_FILES, 108

WriteJPEGFile, 108	WHITE, 315
WriteJPEGMemory, 108	warning, 315
•	YELLOW, 315
keep_var_on_stack	longURL
PassToHTTPThread, 35	URLDB, 40
keepalive	longestStringLength
HTTPHeader, 26	fastStringParser, 22
key	(accounting) (accounting)
hashMapEntry, 24	MAGENTA
keyHash	logs.h, 315
hashMapEntry, 24	MAX_BINDING_PORT
keyLength	ScriptRunner/main.cpp, 322
hashMapEntry, 24	Services/AmmarServer/main.c, 76
	Services/GeoPosShare/main.c, 80
LF	Services/MyURL/main.c, 87
http_header_analysis.c, 188	Services/ScriptRunner/main.cpp, 328
last callback	MAX_CLIENT_THREADS
AmmServer_RH_Context, 18	server_configuration.h, 241
length	MAX_COMMAND_SIZE
tokens, 39	ScriptRunner/main.cpp, 322
links	Services/ScriptRunner/main.cpp, 328
Services/MyURL/main.c, 94	MAX COMPLICITY
LoadConfigurationFile	InputParser_C.h, 211
server_configuration.c, 234	MAX CONTENT TYPE
server_configuration.b, 244	
	server_configuration.h, 241
LoadMyURLDBFile	MAX_FILE_PATH
Services/MyURL/main.c, 90	AmmServerlib.h, 113
loaded_cache_items	MAX_IP_STRING_SIZE
AmmServer_Instance, 15	AmmServerlib.h, 113
loaded_cache_items_Kbytes	MAX_LINKS
AmmServer_Instance, 15	Services/MyURL/main.c, 87
loaded_links	MAX_MEMORY
Services/MyURL/main.c, 95	InputParser_C.h, 211
local_allocation	MAX_QUERY
InputParserC, 34	AmmServerlib.h, 113
logs.c	MAX_RESOURCE
AccessLogAppend, 312	AmmServerlib.h, 113
error, 312	MAX_STRING
ErrorLogAppend, 312	InputParser_C.h, 211
warning, 312	MAX_TO_SIZE
logs.h	Services/MyURL/main.c, 87
AccessLogAppend, 315	MAXIMUM_LEVELS
BLACK, 315	fastStringParser.c, 333
BLUE, 315	MAXcompressedContentSize
BOLDBLACK, 315	AmmServer_DynamicRequest, 14
BOLDBLUE, 315	MAXcontentSize
BOLDCYAN, 315	AmmServer_DynamicRequest, 14
BOLDGREEN, 315	MAXstringsLoaded
BOLDMAGENTA, 315	fastStringParser, 22
BOLDRED, 315	main
BOLDWHITE, 315	AmmCaptcha/AmmCaptchaTester/main.c, 45
BOLDYELLOW, 315	AmmServerlib/InputParser/InputParser_C
CYAN, 315	Tester/main.c, 51
error, 315	helloworld.c, 42
ErrorLogAppend, 315	ScriptRunner/main.cpp, 323
GREEN, 315	Services/AmmarServer/main.c, 77
MAGENTA, 315	Services/GeoPosShare/main.c, 80
NORMAL, 315	Services/MyLoader/main.c, 83
RED, 315	Services/MyURL/main.c, 91
,	55555,, 51 Emains, V1

Services/ScriptRunner/main.cpp, 329	AmmServerlib.h, 113
Services/SimpleTemplate/main.c, 96	POST_request
StringRecognizer/main.c, 98	AmmServer_DynamicRequest, 14
testHashMap.c, 339	POST_request_length
MainHTTPServerThread	AmmServer_DynamicRequest, 14
threadedServer.c, 276	POSTrequest
max_container_count	HTTPHeader, 26
InputParserC, 34	POSTrequestSize
max_delimeter_count	HTTPHeader, 26
InputParserC, 34	PPMREADBUFLEN
max_ret_word	imaging.c, 99
AmmServerlib/InputParser/InputParser_C Tester/main.c, 51	page
maxNumberOfEntries	ScriptRunner/main.cpp, 326
hashMap, 24	Services/ScriptRunner/main.cpp, 332
minute	pageLength ScriptRunner/main.cpp, 326
timestamp, 39	Services/ScriptRunner/main.cpp, 332
modification	parentKeepMessageOnStackUntilReady
cache_item, 20	threadInitHelper.c, 286
month	threadInitHelper.h, 288
timestamp, 39	parentKeepMessageOnStackUntilReadyOrTimeout
months	threadInitHelper.c, 286
time_provider.c, 317	threadInitHelper.h, 288
myurl_server	ParseString
Services/MyURL/main.c, 95	AmmServerlib/InputParser/InputParser_C
	Tester/main.c, 52
NO_FILETYPE	PassToHTTPThread, 34
http_tools.h, 303	client, 35
NONE	clientlen, 35
AmmServerlib.h, 114	clientsock, 35
NORMAL	instance, 35
logs.h, 315	keep_var_on_stack, 35
OPTIONS	port, 36
OPTIONS	pre_spawned_thread, 36
AmmServerlib.h, 114	thread id, 36
outgoingBody HTTPTransaction, 28	PassToPreSpawnedThread, 36
outgoingBodySize	i_adapt, 36
HTTPTransaction, 28	instance, 36
TITTI Hallsaction, 20	path_cat
PATCH	directory_lists.c, 290
AmmServerlib.h, 114	pause_server
POST	AmmServer_Instance, 15
AmmServerlib.h, 114	payload
POSTHEADER_CONTENT_DISPOSITION	hashMapEntry, 24
postHeader.h, 259	payloadLength
POSTHEADER_CONTENT_LENGTH	hashMapEntry, 24
postHeader.h, 259	pixels
POSTHEADER_CONTENT_TYPE	Image, 28
postHeader.h, 259	port
POSTHEADER_EMPTY	PassToHTTPThread, 36
postHeader.h, 259	postHeader.h
POSTHEADER_END_OF_ITEMS	POSTHEADER_CONTENT_DISPOSITION, 259
postHeader.h, 259	POSTHEADER_CONTENT_LENGTH, 259
PUT	POSTHEADER_CONTENT_TYPE, 259
AmmServerlib.h, 114	POSTHEADER_EMPTY, 259
PASSWORD	POSTHEADER_END_OF_ITEMS, 259
AmmServer_Instance_Settings, 16	post_header_analysis.c
POPEN_BUFFER_SIZE	AnalyzePOSTLineRequest, 196

post_header_analysis.h	AmmServer_Instance, 16
AnalyzePOSTLineRequest, 199	prespawnedThreadFlag
postHeader.c	HTTPTransaction, 28
scanFor_postHeader, 258	prespawnedThreads.c
postHeader.h	PreSpawnThreads, 272
scanFor_postHeader, 260	PreSpawnedThread, 271
pre_spawned_thread	UsePreSpawnedThreadToServeNewClient, 272
PassToHTTPThread, 36	prespawnedThreads.h
PreSpawnThreads	PreSpawnThreads, 274
prespawnedThreads.c, 272	UsePreSpawnedThreadToServeNewClient, 275
prespawnedThreads.h, 274	printAllEnumeratorItems
PreSpawnedThread, 37	fastStringParser.c, 335
busy, 38	printlfAllPossibleStrings
client, 38	fastStringParser.c, 335
clientlen, 38	printURLDB
	Services/MyURL/main.c, 92
clientsock, 38	ProcessAuthorizationHTTPLine
instance, 38	http header analysis.c, 190
prespawnedThreads.c, 271	ProcessFirstHTTPLine
templates_root, 38	http_header_analysis.c, 190
thread_id, 38	ProcessRangeHTTPLine
threadNum, 38	http_header_analysis.c, 191
webserver_root, 38	processUploadCallback
prepare_base_image	Services/MyLoader/main.c, 84
ScriptRunner/main.cpp, 324	Services/MyLoader/Main.c, 64
Services/ScriptRunner/main.cpp, 330	RESERVED_CTE_VALUE
prepare_chatbox_content_callback	http_tools.h, 303
Services/AmmarServer/main.c, 77	RC FILEVERSION
prepare_form_content_callback	version.h, 320
ScriptRunner/main.cpp, 324	RED
Services/AmmarServer/main.c, 78	logs.h, 315
Services/ScriptRunner/main.cpp, 330	RH_Scenario
prepare_gps_content_callback	AmmServer_RH_Context, 19
Services/AmmarServer/main.c, 78	RHScenarios
Services/GeoPosShare/main.c, 81	AmmServerlib.h, 114
prepare_helloworld_content_callback	random_chars
helloworld.c, 42	ScriptRunner/main.cpp, 326
prepare_index_content_callback	Services/AmmarServer/main.c, 79
ScriptRunner/main.cpp, 324	Services/ScriptRunner/main.cpp, 332
Services/ScriptRunner/main.cpp, 330	Services/SimpleTemplate/main.c, 97
prepare_random_content_callback	range_end
Services/AmmarServer/main.c, 78	HTTPHeader, 26
Services/SimpleTemplate/main.c, 97	range_start
prepare_stats_content_callback	HTTPHeader, 26
ScriptRunner/main.cpp, 325	ReWriteMyURLDBFile
Services/AmmarServer/main.c, 78	Services/MyURL/main.c, 93
Services/MyLoader/main.c, 84	ReadJPEG
Services/ScriptRunner/main.cpp, 331	jpglnput.c, 105
Services/SimpleTemplate/main.c, 97	jpgInput.h, 108
prepare_top_image	ReadPPM
ScriptRunner/main.cpp, 325	imaging.c, 100
Services/ScriptRunner/main.cpp, 331	imaging.h, 100
prespawn_jobs_finished	ReceiveHTTPHeader
AmmServer_Instance, 15	http_header_analysis.c, 191
prespawn_jobs_started	http_header_analysis.h, 195
AmmServer_Instance, 15	recursiveTraverser
prespawn_turn_to_serve	fastStringParser.c, 336
AmmServer_Instance, 15	ReducePathSlashes_Inplace
prespawned_pool	http_tools.c, 298
	• —

http. toole h. 200	imagoFilos h. 257
http_tools.h, 309 referer	imageFiles.h, 257
HTTPHeader, 26	scanFor_postHeader
	postHeader.c, 258
refererLength HTTPHeader, 26	postHeader.h, 260
	scanFor_textFiles
RenderString AmmCaptcha/main.c, 49	textFiles.b. 262
replaceChar	textFiles.h, 263
ScriptRunner/main.cpp, 325	scanFor_videoFiles
Services/ScriptRunner/main.cpp, 331	videoFiles.c, 264
	videoFiles.h, 266
request AmmServer_RequestOverride_Context, 17	ScriptRunner/main.cpp
request_override_callback	ADMIN_BINDING_PORT, 322
AmmServer_RequestOverride_Context, 17	admin_root, 326
Services/AmmarServer/main.c, 78	admin_server, 326
Services/GeoPosShare/main.c, 81	base_image, 326
Services/MyLoader/main.c, 84	chatbox, 326
Services/SimpleTemplate/main.c, 97	close_dynamic_content, 322
requestContext	default_server, 326
AmmServer RH Context, 18	ENABLE_ADMIN_PAGE, 322
RequestHTTPWebPage	ENABLE_CHAT_BOX, 322
http_tools.c, 299	execute, 322
http_tools.h, 309	form, 326
requestHeader	GET_override, 326
AmmServer_RequestOverride_Context, 17	getBackCommandLine, 322
requestResolver	indexPage, 326
Services/MyURL/main.c, 95	init_dynamic_content, 323
requestType	joystickExecute, 323
HTTPHeader, 26	MAX_BINDING_PORT, 322
resolveRequest	MAX_COMMAND_SIZE, 322
Services/MyURL/main.c, 92	main, 323
ResortDB	page, <mark>326</mark>
Services/MyURL/main.c, 92	pageLength, 326
resource	prepare_base_image, 324
HTTPHeader, 26	prepare_form_content_callback, 324
resource name	prepare_index_content_callback, 324
AmmServer_RH_Context, 18	prepare_stats_content_callback, 325
resourceCacheID	prepare_top_image, 325
HTTPTransaction, 28	random_chars, 326
TTTT Transaction, 20	replaceChar, 325
SAME_PAGE_FOR_ALL_CLIENTS	settings, 326
AmmServerlib.h, 114	stats, 326
saveDynamicRequest	store_new_configuration_callback, 325
dynamic_requests.c, 140	templates_root, 326
dynamic_requests.h, 143	termination_handler, 325
scanFor_applicationFiles	top_image, 326
applicationFiles.c, 246	webserver_root, 326
applicationFiles.h, 248	second
scanFor_audioFiles	timestamp, 39
audioFiles.c, 248	seek_blank_char
audioFiles.h, 250	http_tools.c, 299
scanFor_firstLines	http_tools.h, 309
firstLines.c, 251	seek_non_blank_char
firstLines.h, 252	http_tools.c, 299
scanFor_httpHeader	http_tools.h, 310
httpHeader.c, 253	SendAuthorizationHeader
httpHeader.h, 255	sendHTTPHeader.c, 226
scanFor_imageFiles	sendHTTPHeader.h, 229
imageFiles.c, 256	SendErrorCodeHeader
<del>-</del>	

aandUTTDUaadar a 226	instance CountErcoOP 222
sendHTTPHeader.c, 226	instance_CountFreeOP, 232
sendHTTPHeader.h, 229	instance_CountNewMallocOP, 234
SendErrorFile	instance_WeCanCommitMoreMemory, 234
file_server.c, 218	LoadConfigurationFile, 234
file_server.h, 223	SetUsernameAndPassword, 234
SendFile	TemplatesInternalURI, 236
file_server.c, 218	varSocketTimeoutREAD_seconds, 236
file_server.h, 223	varSocketTimeoutWRITE_seconds, 236
sendHTTPHeader.c	server_configuration.h
SendAuthorizationHeader, 226	AccessLog, 245
SendErrorCodeHeader, 226	AccessLogEnable, 245
SendNotModifiedHeader, 227	AssignStr, 243
SendSuccessCodeHeader, 227	CACHING_ENABLED, 245
sendHTTPHeader.h	CHANGE PRIORITY, 245
SendAuthorizationHeader, 229	CHANGE TO UID, 245
SendErrorCodeHeader, 229	ENABLE POST, 240
SendNotModifiedHeader, 230	EmmitPossibleConfigurationWarnings, 243
SendSuccessCodeHeader, 230	ErrorLog, 245
SendMemoryBlockAsFile	ErrorLogEnable, 245
•	
file_server.c, 219	instance_CountFreeOP, 243
file_server.h, 224	instance_CountNewMallocOP, 243
SendNotModifiedHeader	instance_WeCanCommitMoreMemory, 243
sendHTTPHeader.c, 227	LoadConfigurationFile, 244
sendHTTPHeader.h, 230	MAX_CONTENT_TYPE, 241
SendPart	SetUsernameAndPassword, 244
file_server.c, 220	TemplatesInternalURI, 245
SendSuccessCodeHeader	varSocketTimeoutREAD_seconds, 245
sendHTTPHeader.c, 227	varSocketTimeoutWRITE_seconds, 245
sendHTTPHeader.h, 230	server_running
SeperateWords	AmmServer_Instance, 16
InputParser, 31	server_thread_id
SeperateWordsCC	AmmServer_Instance, 16
InputParser, 32	ServerThreads_DropRootUID
SeperateWordsUC	http_tools.c, 299
InputParser, 32	http_tools.h, 310
serve_captcha_page	serversock
Services/MyURL/main.c, 93	AmmServer Instance, 16
serve create url page	service_filename
Services/MyURL/main.c, 93	Services/MyURL/main.c, 95
serve_error_url_page	service_filename_noslash
Services/MyURL/main.c, 93	Services/MyURL/main.c, 95
serve_goto_url_page	service root
Services/MyURL/main.c, 94	Service_Not Services/MyURL/main.c, 95
ServeClient	service_root_withoutfilename
threadedServer.c, 277	Services/MyURL/main.c, 95
threadedServer.h, 282	Services/AmmarServer/main.c
ServeClientKeepAliveLoop	admin_root, 78
threadedServer.c, 278	admin_server, 78
server_configuration.c	chatbox, 78
AccessLog, 235	close_dynamic_content, 76
AccessLogEnable, 235	default_server, 78
AssignStr, 232	ENABLE_CHAT_BOX, 76
CACHING_ENABLED, 235	executeScript, 78
CHANGE_PRIORITY, 235	executeScriptFunction, 76
CHANGE_TO_UID, 235	executeScriptRC, 79
EmmitPossibleConfigurationWarnings, 232	form, 79
ErrorLog, 235	GET_override, 79
ErrorLogEnable, 235	gps, 79

init_dynamic_content, 76	indexPagePath, 94
MAX_BINDING_PORT, 76	init_dynamic_content, 90
main, 77	is_an_unsafe_str, 90
prepare_chatbox_content_callback, 77	isURLDBSorted, 90
prepare_form_content_callback, 78	links, 94
prepare_gps_content_callback, 78	LoadMyURLDBFile, 90
prepare_random_content_callback, 78	loaded_links, 95
prepare_stats_content_callback, 78	MAX_LINKS, 87
random_chars, 79	MAX_TO_SIZE, 87
request_override_callback, 78	main, 91
stats, 79	myurl_server, 95
templates root, 79	printURLDB, 92
• – •	•
webserver_root, 79	ReWriteMyURLDBFile, 93
Services/GeoPosShare/main.c	requestResolver, 95
admin_root, 81	resolveRequest, 92
close_dynamic_content, 80	ResortDB, 92
default_server, 81	serve_captcha_page, 93
GET_override, 82	serve_create_url_page, 93
gps, 82	serve_error_url_page, 93
init_dynamic_content, 80	serve_goto_url_page, 94
main, 80	service_filename, 95
prepare_gps_content_callback, 81	service_filename_noslash, 95
request_override_callback, 81	service_root, 95
templates_root, 82	service_root_withoutfilename, 95
webserver_root, 82	sorted_links, 95
Services/MyLoader/main.c	struct_cmp_urldb_items, 94
close_dynamic_content, 83	templates root, 95
default_server, 85	webserver_root, 95
GET_override, 85	Services/ScriptRunner/main.cpp
init_dynamic_content, 83	admin_root, 332
— · —	
main, 83	admin_server, 332
prepare_stats_content_callback, 84	base_image, 332
processUploadCallback, 84	chatbox, 332
request_override_callback, 84	close_dynamic_content, 328
stats, 85	default_server, 332
templates_root, 85	ENABLE_CHAT_BOX, 328
uploadProcessor, 85	EraseFile, 328
webserver_root, 85	execute, 328
Services/MyURL/main.c	FileExistsTest, 329
Add_MyURL, 87	form, 332
allocateLinksIfNeeded, 88	GET_override, 332
allocated_links, 94	getBackCommandLine, 329
Append2MyURLDBFile, 88	indexPage, 332
captcha_url, 94	init_dynamic_content, 329
close_dynamic_content, 88	joystickExecute, 329
create_url, 94	MAX_BINDING_PORT, 328
db addIDLock, 94	MAX COMMAND SIZE, 328
db_file, 94	main, 329
db_fileLock, 94	page, 332
default failed, 94	pageLength, 332
error_url, 94	prepare_base_image, 330
Find_longURL, 88	prepare_form_content_callback, 330
Find_longURLSerial, 89	prepare_index_content_callback, 330
Get_longURL, 89	– – –
_ •	prepare_stats_content_callback, 331
goto_url, 94	prepare_top_image, 331
hashURL, 89	random_chars, 332
indexPage, 94	replaceChar, 331
indexPageLength, 94	settings, 332

stats, 332	src/AmmServerlib/InputParser/InputParser_C_Tester/main
store_new_configuration_callback, 331	c, 50
StringIsHTMLSafe, 331	src/AmmServerlib/cache/client_list.c, 131
templates_root, 332	src/AmmServerlib/cache/client_list.h, 135
termination_handler, 331	src/AmmServerlib/cache/dynamic_requests.c, 138
top_image, 332	src/AmmServerlib/cache/dynamic_requests.h, 140
webserver_root, 332	src/AmmServerlib/cache/file_caching.c, 143
Services/SimpleTemplate/main.c	src/AmmServerlib/cache/file_caching.h, 151
close_dynamic_content, 96	src/AmmServerlib/cache/file_compression.c, 159
default server, 97	src/AmmServerlib/cache/file_compression.h, 162
GET_override, 97	src/AmmServerlib/hashmap/hashmap.c, 164
init_dynamic_content, 96	src/AmmServerlib/hashmap/hashmap.h, 176
main, 96	src/AmmServerlib/header_analysis/http_header
prepare_random_content_callback, 97	analysis.c, 187
prepare_stats_content_callback, 97	src/AmmServerlib/header_analysis/http_header
random_chars, 97	analysis.h, 192
request_override_callback, 97	src/AmmServerlib/header_analysis/post_header
stats, 97	analysis.c, 196
templates_root, 97	src/AmmServerlib/header_analysis/post_header
webserver root, 97	analysis.h, 198
SetDelimeter	src/AmmServerlib/main.c, 53
InputParser, 32	src/AmmServerlib/network/file_server.c, 217
setSocketTimeouts	src/AmmServerlib/network/file_server.h, 221
http tools.c, 299	src/AmmServerlib/network/sendHTTPHeader.c, 225
http_tools.h, 310	src/AmmServerlib/network/sendHTTPHeader.h, 228
SetUsernameAndPassword	src/AmmServerlib/server_configuration.c, 231
server_configuration.c, 234	src/AmmServerlib/server_configuration.h, 236
server_configuration.h, 244	src/AmmServerlib/stringscanners/applicationFiles.c,
	246
settings	
settings AmmServer Instance 16	src/Amm Server lib/string scanners/application Files.h,
AmmServer_Instance, 16	246
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326	246 src/AmmServerlib/stringscanners/audioFiles.c, 248
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332	246 src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.h, 249
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332 shortURL	246 src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.h, 249 src/AmmServerlib/stringscanners/firstLines.c, 250
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332 shortURL URLDB, 40	246 src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.h, 249 src/AmmServerlib/stringscanners/firstLines.c, 250 src/AmmServerlib/stringscanners/firstLines.h, 251
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332 shortURL URLDB, 40 shortURLHash	246 src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.h, 249 src/AmmServerlib/stringscanners/firstLines.c, 250 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/httpHeader.c, 252
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332 shortURL URLDB, 40 shortURLHash URLDB, 40	src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.h, 249 src/AmmServerlib/stringscanners/firstLines.c, 250 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/httpHeader.c, 252 src/AmmServerlib/stringscanners/httpHeader.h, 253
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332 shortURL URLDB, 40 shortURLHash URLDB, 40 shortestStringLength	src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.h, 249 src/AmmServerlib/stringscanners/firstLines.c, 250 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/httpHeader.c, 252 src/AmmServerlib/stringscanners/httpHeader.h, 253 src/AmmServerlib/stringscanners/imageFiles.c, 255
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332 ShortURL URLDB, 40 ShortURLHash URLDB, 40 ShortestStringLength fastStringParser, 22	src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.h, 249 src/AmmServerlib/stringscanners/firstLines.c, 250 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/httpHeader.c, 252 src/AmmServerlib/stringscanners/httpHeader.h, 253 src/AmmServerlib/stringscanners/imageFiles.c, 255 src/AmmServerlib/stringscanners/imageFiles.h, 256
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332 shortURL URLDB, 40 shortURLHash URLDB, 40 shortestStringLength fastStringParser, 22 sorted_links	src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.h, 249 src/AmmServerlib/stringscanners/firstLines.c, 250 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/fitpHeader.c, 252 src/AmmServerlib/stringscanners/httpHeader.h, 253 src/AmmServerlib/stringscanners/imageFiles.c, 255 src/AmmServerlib/stringscanners/imageFiles.h, 256 src/AmmServerlib/stringscanners/postHeader.c, 257
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332 shortURL URLDB, 40 shortURLHash URLDB, 40 shortestStringLength fastStringParser, 22 sorted_links Services/MyURL/main.c, 95	src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.h, 249 src/AmmServerlib/stringscanners/firstLines.c, 250 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/httpHeader.c, 252 src/AmmServerlib/stringscanners/imageFiles.c, 255 src/AmmServerlib/stringscanners/imageFiles.h, 256 src/AmmServerlib/stringscanners/postHeader.c, 257 src/AmmServerlib/stringscanners/postHeader.h, 258
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332 ShortURL URLDB, 40 ShortURLHash URLDB, 40 ShortestStringLength fastStringParser, 22 Sorted_links Services/MyURL/main.c, 95 SpawnThreadToServeNewClient	src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.h, 249 src/AmmServerlib/stringscanners/firstLines.c, 250 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/httpHeader.c, 252 src/AmmServerlib/stringscanners/httpHeader.h, 253 src/AmmServerlib/stringscanners/imageFiles.c, 255 src/AmmServerlib/stringscanners/imageFiles.h, 256 src/AmmServerlib/stringscanners/postHeader.c, 257 src/AmmServerlib/stringscanners/postHeader.h, 258 src/AmmServerlib/stringscanners/textFiles.c, 261
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332 ShortURL URLDB, 40 ShortURLHash URLDB, 40 ShortestStringLength fastStringParser, 22 Sorted_links Services/MyURL/main.c, 95 SpawnThreadToServeNewClient freshThreads.c, 267	src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.h, 249 src/AmmServerlib/stringscanners/firstLines.c, 250 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/httpHeader.c, 252 src/AmmServerlib/stringscanners/httpHeader.h, 253 src/AmmServerlib/stringscanners/imageFiles.c, 255 src/AmmServerlib/stringscanners/imageFiles.h, 256 src/AmmServerlib/stringscanners/postHeader.c, 257 src/AmmServerlib/stringscanners/postHeader.h, 258 src/AmmServerlib/stringscanners/textFiles.c, 261 src/AmmServerlib/stringscanners/textFiles.h, 262
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332 ShortURL URLDB, 40 ShortURLHash URLDB, 40 ShortestStringLength fastStringParser, 22 Sorted_links Services/MyURL/main.c, 95 SpawnThreadToServeNewClient freshThreads.c, 267 freshThreads.h, 270	src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.h, 249 src/AmmServerlib/stringscanners/firstLines.c, 250 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/httpHeader.c, 252 src/AmmServerlib/stringscanners/imageFiles.c, 253 src/AmmServerlib/stringscanners/imageFiles.c, 255 src/AmmServerlib/stringscanners/imageFiles.h, 256 src/AmmServerlib/stringscanners/postHeader.c, 257 src/AmmServerlib/stringscanners/postHeader.h, 258 src/AmmServerlib/stringscanners/textFiles.c, 261 src/AmmServerlib/stringscanners/textFiles.h, 262 src/AmmServerlib/stringscanners/videoFiles.c, 263
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332 ShortURL URLDB, 40 ShortURLHash URLDB, 40 ShortestStringLength fastStringParser, 22 Sorted_links Services/MyURL/main.c, 95 SpawnThreadToServeNewClient freshThreads.c, 267 freshThreads.h, 270 src/AmmCaptcha/AmmCaptcha.h, 43	src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.h, 249 src/AmmServerlib/stringscanners/firstLines.c, 250 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/fitpHeader.c, 252 src/AmmServerlib/stringscanners/httpHeader.h, 253 src/AmmServerlib/stringscanners/imageFiles.c, 255 src/AmmServerlib/stringscanners/imageFiles.h, 256 src/AmmServerlib/stringscanners/postHeader.c, 257 src/AmmServerlib/stringscanners/postHeader.h, 258 src/AmmServerlib/stringscanners/textFiles.c, 261 src/AmmServerlib/stringscanners/textFiles.h, 262 src/AmmServerlib/stringscanners/videoFiles.c, 263 src/AmmServerlib/stringscanners/videoFiles.h, 265
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332 shortURL URLDB, 40 shortURLHash URLDB, 40 shortestStringLength fastStringParser, 22 sorted_links Services/MyURL/main.c, 95 SpawnThreadToServeNewClient freshThreads.c, 267 freshThreads.h, 270 src/AmmCaptcha/AmmCaptcha.h, 43 src/AmmCaptcha/AmmCaptchaTester/main.c, 45	src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.c, 249 src/AmmServerlib/stringscanners/firstLines.c, 250 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/fitpHeader.c, 252 src/AmmServerlib/stringscanners/httpHeader.h, 253 src/AmmServerlib/stringscanners/imageFiles.c, 255 src/AmmServerlib/stringscanners/imageFiles.h, 256 src/AmmServerlib/stringscanners/postHeader.c, 257 src/AmmServerlib/stringscanners/postHeader.h, 258 src/AmmServerlib/stringscanners/textFiles.c, 261 src/AmmServerlib/stringscanners/textFiles.h, 262 src/AmmServerlib/stringscanners/videoFiles.c, 263 src/AmmServerlib/stringscanners/videoFiles.h, 265 src/AmmServerlib/stringscanners/videoFiles.h, 265 src/AmmServerlib/threads/freshThreads.c, 266
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332 ShortURL URLDB, 40 ShortURLHash URLDB, 40 ShortestStringLength fastStringParser, 22 Sorted_links Services/MyURL/main.c, 95 SpawnThreadToServeNewClient freshThreads.c, 267 freshThreads.h, 270 Src/AmmCaptcha/AmmCaptcha.h, 43 Src/AmmCaptcha/AmmCaptchaTester/main.c, 45 Src/AmmCaptcha/imaging.c, 98	src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.h, 249 src/AmmServerlib/stringscanners/firstLines.c, 250 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/httpHeader.c, 252 src/AmmServerlib/stringscanners/imageFiles.c, 255 src/AmmServerlib/stringscanners/imageFiles.c, 255 src/AmmServerlib/stringscanners/postHeader.c, 257 src/AmmServerlib/stringscanners/postHeader.c, 257 src/AmmServerlib/stringscanners/postHeader.h, 258 src/AmmServerlib/stringscanners/textFiles.c, 261 src/AmmServerlib/stringscanners/videoFiles.c, 263 src/AmmServerlib/stringscanners/videoFiles.h, 265 src/AmmServerlib/threads/freshThreads.c, 266 src/AmmServerlib/threads/freshThreads.h, 268
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332 ShortURL URLDB, 40 ShortURLHash URLDB, 40 ShortestStringLength fastStringParser, 22 Sorted_links Services/MyURL/main.c, 95 SpawnThreadToServeNewClient freshThreads.c, 267 freshThreads.h, 270 src/AmmCaptcha/AmmCaptcha.h, 43 src/AmmCaptcha/imaging.c, 98 src/AmmCaptcha/imaging.h, 100	src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.h, 249 src/AmmServerlib/stringscanners/firstLines.c, 250 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/httpHeader.c, 252 src/AmmServerlib/stringscanners/imageFiles.c, 253 src/AmmServerlib/stringscanners/imageFiles.c, 255 src/AmmServerlib/stringscanners/jostHeader.c, 257 src/AmmServerlib/stringscanners/postHeader.c, 257 src/AmmServerlib/stringscanners/postHeader.h, 258 src/AmmServerlib/stringscanners/textFiles.c, 261 src/AmmServerlib/stringscanners/textFiles.h, 262 src/AmmServerlib/stringscanners/videoFiles.h, 265 src/AmmServerlib/stringscanners/videoFiles.h, 265 src/AmmServerlib/threads/freshThreads.c, 266 src/AmmServerlib/threads/freshThreads.h, 268 src/AmmServerlib/threads/prespawnedThreads.c, 271
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332 ShortURL URLDB, 40 ShortURLHash URLDB, 40 ShortestStringLength fastStringParser, 22 Sorted_links Services/MyURL/main.c, 95 SpawnThreadToServeNewClient freshThreads.c, 267 freshThreads.h, 270 src/AmmCaptcha/AmmCaptcha.h, 43 src/AmmCaptcha/Imaging.c, 98 src/AmmCaptcha/imaging.h, 100 src/AmmCaptcha/img_warp.c, 101	src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.h, 249 src/AmmServerlib/stringscanners/firstLines.c, 250 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/httpHeader.c, 252 src/AmmServerlib/stringscanners/imageFiles.c, 255 src/AmmServerlib/stringscanners/imageFiles.c, 255 src/AmmServerlib/stringscanners/imageFiles.h, 256 src/AmmServerlib/stringscanners/postHeader.c, 257 src/AmmServerlib/stringscanners/postHeader.h, 258 src/AmmServerlib/stringscanners/textFiles.c, 261 src/AmmServerlib/stringscanners/videoFiles.h, 262 src/AmmServerlib/stringscanners/videoFiles.h, 265 src/AmmServerlib/threads/freshThreads.c, 266 src/AmmServerlib/threads/freshThreads.h, 268 src/AmmServerlib/threads/prespawnedThreads.c, 271 src/AmmServerlib/threads/prespawnedThreads.h, 273
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332 ShortURL URLDB, 40 ShortURLHash URLDB, 40 ShortestStringLength fastStringParser, 22 Sorted_links Services/MyURL/main.c, 95 SpawnThreadToServeNewClient freshThreads.c, 267 freshThreads.h, 270 src/AmmCaptcha/AmmCaptcha.h, 43 src/AmmCaptcha/AmmCaptchaTester/main.c, 45 src/AmmCaptcha/imaging.c, 98 src/AmmCaptcha/imaging.h, 100 src/AmmCaptcha/img_warp.c, 101 src/AmmCaptcha/img_warp.h, 102	src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.h, 249 src/AmmServerlib/stringscanners/firstLines.c, 250 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/fitpHeader.c, 252 src/AmmServerlib/stringscanners/httpHeader.h, 253 src/AmmServerlib/stringscanners/imageFiles.c, 255 src/AmmServerlib/stringscanners/imageFiles.h, 256 src/AmmServerlib/stringscanners/postHeader.c, 257 src/AmmServerlib/stringscanners/postHeader.h, 258 src/AmmServerlib/stringscanners/textFiles.c, 261 src/AmmServerlib/stringscanners/textFiles.h, 262 src/AmmServerlib/stringscanners/videoFiles.h, 265 src/AmmServerlib/stringscanners/videoFiles.h, 265 src/AmmServerlib/threads/freshThreads.c, 266 src/AmmServerlib/threads/freshThreads.h, 268 src/AmmServerlib/threads/prespawnedThreads.h, 273 src/AmmServerlib/threads/threadInitHelper.c, 285
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332 ShortURL URLDB, 40 ShortURLHash URLDB, 40 ShortestStringLength fastStringParser, 22 Sorted_links Services/MyURL/main.c, 95 SpawnThreadToServeNewClient freshThreads.c, 267 freshThreads.h, 270 src/AmmCaptcha/AmmCaptcha.h, 43 src/AmmCaptcha/AmmCaptchaTester/main.c, 45 src/AmmCaptcha/imaging.c, 98 src/AmmCaptcha/imaging.h, 100 src/AmmCaptcha/img_warp.c, 101 src/AmmCaptcha/img_warp.h, 102 src/AmmCaptcha/jpgInput.c, 104	src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.h, 249 src/AmmServerlib/stringscanners/firstLines.c, 250 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/httpHeader.c, 252 src/AmmServerlib/stringscanners/imageFiles.c, 255 src/AmmServerlib/stringscanners/imageFiles.h, 256 src/AmmServerlib/stringscanners/jostHeader.c, 257 src/AmmServerlib/stringscanners/postHeader.c, 257 src/AmmServerlib/stringscanners/postHeader.h, 258 src/AmmServerlib/stringscanners/textFiles.c, 261 src/AmmServerlib/stringscanners/textFiles.h, 262 src/AmmServerlib/stringscanners/videoFiles.c, 263 src/AmmServerlib/stringscanners/videoFiles.h, 265 src/AmmServerlib/threads/freshThreads.c, 266 src/AmmServerlib/threads/freshThreads.h, 268 src/AmmServerlib/threads/prespawnedThreads.c, 271 src/AmmServerlib/threads/prespawnedThreads.h, 273 src/AmmServerlib/threads/threadInitHelper.c, 285 src/AmmServerlib/threads/threadInitHelper.h, 286
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332 ShortURL URLDB, 40 ShortURLHash URLDB, 40 ShortestStringLength fastStringParser, 22 Sorted_links Services/MyURL/main.c, 95 SpawnThreadToServeNewClient freshThreads.c, 267 freshThreads.h, 270 src/AmmCaptcha/AmmCaptcha.h, 43 src/AmmCaptcha/imaging.c, 98 src/AmmCaptcha/imaging.h, 100 src/AmmCaptcha/img_warp.c, 101 src/AmmCaptcha/jpgInput.c, 104 src/AmmCaptcha/jpgInput.h, 107	src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.h, 249 src/AmmServerlib/stringscanners/firstLines.c, 250 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/httpHeader.c, 252 src/AmmServerlib/stringscanners/httpHeader.h, 253 src/AmmServerlib/stringscanners/imageFiles.c, 255 src/AmmServerlib/stringscanners/imageFiles.h, 256 src/AmmServerlib/stringscanners/postHeader.c, 257 src/AmmServerlib/stringscanners/postHeader.c, 257 src/AmmServerlib/stringscanners/postHeader.h, 258 src/AmmServerlib/stringscanners/textFiles.c, 261 src/AmmServerlib/stringscanners/textFiles.h, 262 src/AmmServerlib/stringscanners/videoFiles.c, 263 src/AmmServerlib/stringscanners/videoFiles.h, 265 src/AmmServerlib/threads/freshThreads.c, 266 src/AmmServerlib/threads/frespawnedThreads.c, 271 src/AmmServerlib/threads/prespawnedThreads.h, 273 src/AmmServerlib/threads/threadInitHelper.c, 285 src/AmmServerlib/threads/threadInitHelper.h, 286 src/AmmServerlib/threads/threadInitHelper.h, 286 src/AmmServerlib/threads/threadedServer.c, 275
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332 ShortURL URLDB, 40 ShortURLHash URLDB, 40 ShortestStringLength fastStringParser, 22 Sorted_links Services/MyURL/main.c, 95 SpawnThreadToServeNewClient freshThreads.c, 267 freshThreads.h, 270 Src/AmmCaptcha/AmmCaptcha-Inster/main.c, 45 Src/AmmCaptcha/imaging.c, 98 src/AmmCaptcha/imaging.h, 100 Src/AmmCaptcha/img_warp.c, 101 src/AmmCaptcha/jpglnput.c, 104 Src/AmmCaptcha/jpglnput.h, 107 Src/AmmCaptcha/jpglnput.h, 107 Src/AmmCaptcha/jpglnput.h, 107 Src/AmmCaptcha/jpglnput.h, 107 Src/AmmCaptcha/main.c, 46	src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.h, 249 src/AmmServerlib/stringscanners/firstLines.c, 250 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/httpHeader.c, 252 src/AmmServerlib/stringscanners/httpHeader.h, 253 src/AmmServerlib/stringscanners/imageFiles.c, 255 src/AmmServerlib/stringscanners/imageFiles.h, 256 src/AmmServerlib/stringscanners/postHeader.c, 257 src/AmmServerlib/stringscanners/postHeader.c, 257 src/AmmServerlib/stringscanners/postHeader.h, 258 src/AmmServerlib/stringscanners/textFiles.c, 261 src/AmmServerlib/stringscanners/textFiles.h, 262 src/AmmServerlib/stringscanners/videoFiles.c, 263 src/AmmServerlib/stringscanners/videoFiles.h, 265 src/AmmServerlib/threads/freshThreads.c, 266 src/AmmServerlib/threads/freshThreads.h, 268 src/AmmServerlib/threads/prespawnedThreads.c, 271 src/AmmServerlib/threads/threadInitHelper.c, 285 src/AmmServerlib/threads/threadInitHelper.h, 286 src/AmmServerlib/threads/threadedServer.c, 275 src/AmmServerlib/threads/threadedServer.h, 281
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332 ShortURL URLDB, 40 ShortURLHash URLDB, 40 ShortestStringLength fastStringParser, 22 Sorted_links Services/MyURL/main.c, 95 SpawnThreadToServeNewClient freshThreads.c, 267 freshThreads.h, 270 Src/AmmCaptcha/AmmCaptcha.h, 43 Src/AmmCaptcha/Imaging.c, 98 Src/AmmCaptcha/imaging.c, 98 Src/AmmCaptcha/img_warp.c, 101 Src/AmmCaptcha/img_warp.c, 101 Src/AmmCaptcha/jpglnput.c, 104 Src/AmmCaptcha/jpglnput.h, 107 Src/AmmCaptcha/main.c, 46 Src/AmmCaptcha/main.c, 46 Src/AmmServerlib/AmmServerlib.h, 109	src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.h, 249 src/AmmServerlib/stringscanners/firstLines.c, 250 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/httpHeader.c, 252 src/AmmServerlib/stringscanners/imageFiles.c, 253 src/AmmServerlib/stringscanners/imageFiles.c, 255 src/AmmServerlib/stringscanners/jostHeader.c, 257 src/AmmServerlib/stringscanners/postHeader.c, 257 src/AmmServerlib/stringscanners/postHeader.h, 258 src/AmmServerlib/stringscanners/textFiles.c, 261 src/AmmServerlib/stringscanners/videoFiles.c, 263 src/AmmServerlib/stringscanners/videoFiles.h, 262 src/AmmServerlib/stringscanners/videoFiles.c, 263 src/AmmServerlib/threads/freshThreads.c, 266 src/AmmServerlib/threads/frespawnedThreads.h, 268 src/AmmServerlib/threads/prespawnedThreads.h, 273 src/AmmServerlib/threads/threadInitHelper.c, 285 src/AmmServerlib/threads/threadedServer.c, 275 src/AmmServerlib/threads/threadedServer.h, 281 src/AmmServerlib/tools/directory_lists.c, 288
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332 ShortURL URLDB, 40 ShortURLHash URLDB, 40 ShortestStringLength fastStringParser, 22 Sorted_links Services/MyURL/main.c, 95 SpawnThreadToServeNewClient freshThreads.c, 267 freshThreads.h, 270 src/AmmCaptcha/AmmCaptcha.h, 43 src/AmmCaptcha/AmmCaptchaTester/main.c, 45 src/AmmCaptcha/imaging.c, 98 src/AmmCaptcha/imaging.h, 100 src/AmmCaptcha/img_warp.c, 101 src/AmmCaptcha/img_warp.h, 102 src/AmmCaptcha/jpgInput.c, 104 src/AmmCaptcha/jpgInput.h, 107 src/AmmCaptcha/main.c, 46 src/AmmServerlib/AmmServerlib.h, 109 src/AmmServerlib/InputParser/InputParser.cpp, 200	src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.h, 249 src/AmmServerlib/stringscanners/firstLines.c, 250 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/httpHeader.c, 252 src/AmmServerlib/stringscanners/httpHeader.h, 253 src/AmmServerlib/stringscanners/imageFiles.c, 255 src/AmmServerlib/stringscanners/imageFiles.h, 256 src/AmmServerlib/stringscanners/postHeader.c, 257 src/AmmServerlib/stringscanners/postHeader.h, 258 src/AmmServerlib/stringscanners/textFiles.c, 261 src/AmmServerlib/stringscanners/textFiles.h, 262 src/AmmServerlib/stringscanners/videoFiles.h, 265 src/AmmServerlib/stringscanners/videoFiles.h, 265 src/AmmServerlib/threads/freshThreads.c, 266 src/AmmServerlib/threads/freshThreads.h, 268 src/AmmServerlib/threads/prespawnedThreads.c, 271 src/AmmServerlib/threads/prespawnedThreads.h, 273 src/AmmServerlib/threads/threadInitHelper.c, 285 src/AmmServerlib/threads/threadedServer.c, 275 src/AmmServerlib/threads/threadedServer.h, 281 src/AmmServerlib/threads/threadedServer.h, 281 src/AmmServerlib/tools/directory_lists.h, 290
AmmServer_Instance, 16 ScriptRunner/main.cpp, 326 Services/ScriptRunner/main.cpp, 332 ShortURL URLDB, 40 ShortURLHash URLDB, 40 ShortestStringLength fastStringParser, 22 Sorted_links Services/MyURL/main.c, 95 SpawnThreadToServeNewClient freshThreads.c, 267 freshThreads.h, 270 Src/AmmCaptcha/AmmCaptcha.h, 43 Src/AmmCaptcha/Imaging.c, 98 Src/AmmCaptcha/imaging.c, 98 Src/AmmCaptcha/img_warp.c, 101 Src/AmmCaptcha/img_warp.c, 101 Src/AmmCaptcha/jpglnput.c, 104 Src/AmmCaptcha/jpglnput.h, 107 Src/AmmCaptcha/main.c, 46 Src/AmmCaptcha/main.c, 46 Src/AmmServerlib/AmmServerlib.h, 109	src/AmmServerlib/stringscanners/audioFiles.c, 248 src/AmmServerlib/stringscanners/audioFiles.h, 249 src/AmmServerlib/stringscanners/firstLines.c, 250 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/firstLines.h, 251 src/AmmServerlib/stringscanners/httpHeader.c, 252 src/AmmServerlib/stringscanners/imageFiles.c, 253 src/AmmServerlib/stringscanners/imageFiles.c, 255 src/AmmServerlib/stringscanners/jostHeader.c, 257 src/AmmServerlib/stringscanners/postHeader.c, 257 src/AmmServerlib/stringscanners/postHeader.h, 258 src/AmmServerlib/stringscanners/textFiles.c, 261 src/AmmServerlib/stringscanners/videoFiles.c, 263 src/AmmServerlib/stringscanners/videoFiles.h, 262 src/AmmServerlib/stringscanners/videoFiles.c, 263 src/AmmServerlib/threads/freshThreads.c, 266 src/AmmServerlib/threads/frespawnedThreads.h, 268 src/AmmServerlib/threads/prespawnedThreads.h, 273 src/AmmServerlib/threads/threadInitHelper.c, 285 src/AmmServerlib/threads/threadedServer.c, 275 src/AmmServerlib/threads/threadedServer.h, 281 src/AmmServerlib/tools/directory_lists.c, 288

src/AmmServerlib/tools/logs.h, 314	http_tools.c, 300
src/AmmServerlib/tools/time_provider.c, 316	http_tools.h, 310
src/AmmServerlib/tools/time_provider.h, 318	StripHTMLCharacters_Inplace
src/AmmServerlib/version.h, 320	http_tools.c, 300
src/ScriptRunner/main.cpp, 320	http_tools.h, 310
	StripVariableFromGETorPOSTString
src/Services/AmmarServer/main.c, 74	-
src/Services/GeoPosShare/main.c, 79	http_tools.c, 300
src/Services/MyLoader/main.c, 82	http_tools.h, 311
src/Services/MyURL/main.c, 85	stristr
src/Services/ScriptRunner/main.cpp, 326	http_tools.c, 300
src/Services/SimpleTemplate/main.c, 95	stristr2Caps
src/StringRecognizer/fastStringParser.c, 332	http_tools.c, 301
src/StringRecognizer/fastStringParser.h, 336	struct_cmp_urldb_items
src/StringRecognizer/main.c, 98	Services/MyURL/main.c, 94
src/UnitTests/testHashMap.c, 338	supports_compression
start_timer	HTTPHeader, 26
time_provider.c, 317	TEVT
time provider.h, 319	TEXT
StartHTTPServer	http_tools.h, 303
threadedServer.c, 279	TEXTFILES_CSS
threadedServer.h, 283	textFiles.h, 262
starting	TEXTFILES_DOC
directory_lists.c, 289	textFiles.h, 263
	TEXTFILES_EMPTY
stats	textFiles.h, 262
ScriptRunner/main.cpp, 326	TEXTFILES_END_OF_ITEMS
Services/AmmarServer/main.c, 79	textFiles.h, 263
Services/MyLoader/main.c, 85	TEXTFILES_HTM
Services/ScriptRunner/main.cpp, 332	textFiles.h, 262
Services/SimpleTemplate/main.c, 97	TEXTFILES HTML
stop_server	textFiles.h, 262
AmmServer_Instance, 16	TEXTFILES ODF
StopHTTPServer	textFiles.h, 263
threadedServer.c, 280	TEXTFILES ODT
threadedServer.h, 284	textFiles.h, 263
store_new_configuration_callback	TEXTFILES_RTF
ScriptRunner/main.cpp, 325	textFiles.h, 263
Services/ScriptRunner/main.cpp, 331	TEXTFILES_TXT
str	textFiles.h, 262
fspString, 22	TRACE
InputParserC, 34	AmmServerlib.h, 114
Str2Int internal	
InputParser C.c, 209	tag_after_image
str_length	directory_lists.c, 289
InputParserC, 34	tag_pre_image
strIDFriendly	directory_lists.c, 289
fspString, 22	templates_root
	AmmServer_Instance, 16
strLength	helloworld.c, 43
fspString, 22	PreSpawnedThread, 38
strToUpcase	ScriptRunner/main.cpp, 326
http_tools.c, 301	Services/AmmarServer/main.c, 79
http_tools.h, 311	Services/GeoPosShare/main.c, 82
StringIsHTMLSafe	Services/MyLoader/main.c, 85
Services/ScriptRunner/main.cpp, 331	Services/MyURL/main.c, 95
StringRecognizer/main.c	Services/ScriptRunner/main.cpp, 332
main, 98	Services/SimpleTemplate/main.c, 97
stringsLoaded	TemplatesInternalURI
fastStringParser, 22	server_configuration.c, 236
StripGETRequestQueryAndFragment	server_configuration.h, 245

term_buffer	time_provider.c
jpgInput.c, 105	days, 317
termination_handler	end_timer, 316
ScriptRunner/main.cpp, 325	GetDateString, 317
Services/ScriptRunner/main.cpp, 331	GetTickCountAmmServ, 317
TerminationCallback	months, 317
AmmServerlib/main.c, 74	start_timer, 317
testAmmCaptcha	time_provider.h
AmmCaptcha.h, 44	end_timer, 319
AmmCaptcha/main.c, 49	GetDateString, 319
testHashMap.c	GetTickCountAmmServ, 319
main, 339	start_timer, 319
textFiles.h	time_snap, 38
TEXTFILES_CSS, 262	difference, 38
TEXTFILES_DOC, 263	timestamp, 38
TEXTFILES_EMPTY, 262	day, 39
TEXTFILES_END_OF_ITEMS, 263	hour, 39
TEXTFILES HTM, 262	minute, 39
TEXTFILES_HTML, 262	month, 39
TEXTFILES ODF, 263	second, 39
TEXTFILES_ODT, 263	wday, 39
TEXTFILES RTF, 263	year, 39
TEXTFILES TXT, 262	token_start
textFiles.c	tokens, 39
scanFor_textFiles, 261	tokenlist
textFiles.h	InputParserC, 34
scanFor_textFiles, 263	tokens, 39
thread_id	length, 39
PassToHTTPThread, 36	token_start, 39
PreSpawnedThread, 38	tokens_count
threadID	InputParserC, 34
HTTPTransaction, 28	tokens_max
threadInitHelper.c	InputParserC, 34
childFinishedWithParentMessage, 285	top_image
parentKeepMessageOnStackUntilReady, 286	ScriptRunner/main.cpp, 326
parentKeepMessageOnStackUntilReadyOr-	Services/ScriptRunner/main.cpp, 332
Timeout, 286	TransmitFileToSocket
threadInitHelper.h	file_server.c, 220
childFinishedWithParentMessage, 287	TransmitFileToSocketInternal
parentKeepMessageOnStackUntilReady, 288	file_server.c, 220
parentKeepMessageOnStackUntilReadyOr-	trim_last_empty_chars
Timeout, 288	http_tools.c, 301
threadNum	http_tools.h, 311
PreSpawnedThread, 38	TypesOfRequests
threadedServer.c	AmmServerlib.h, 114
HTTPServerIsRunning, 276	LIDLDD 40
MainHTTPServerThread, 276	URLDB, 40
ServeClient, 277	longURL, 40
ServeClientKeepAliveLoop, 278	shortURL, 40
StartHTTPServer, 279	shortURLHash, 40
StopHTTPServer, 280	USE_BINARY_SEARCH
threadedServer.h	Services/MyURL/main.c, 87
HTTPServerIsRunning, 282	USE_JPG_FILES
ServeClient, 282	jpgInput.h, 108 USERNAME
StartHTTPServer, 284	AmmServer_Instance_Settings, 16
StopHTTPServer, 284	uploadProcessor
threads_pool AmmServer_Instance, 16	Services/MyLoader/main.c, 85 UsePreSpawnedThreadToServeNewClient
AHIHOGIVGI_HISIAHOG, TO	Oser respanned i inteau roser vernemonent

prespawnedThreads.c, 272 prespawnedThreads.h, 275 userAgent HTTPHeader, 26 userAgentLength HTTPHeader, 26	VIDEOFILES_MP4, 266 VIDEOFILES_MPEG, 266 VIDEOFILES_MPEG4, 266 VIDEOFILES_WEBM, 266 videoFiles.c scanFor_videoFiles, 264
userList clientListContext, 21	videoFiles.h scanFor_videoFiles, 266
VIDEO	WHITE
http_tools.h, 303	logs.h, 315
VIDEOFILES 3GP	warning
videoFiles.h, 266	logs.c, 312
VIDEOFILES_AVI	logs.h, 315
videoFiles.h, 266	warningsAboutIncorrectlyAllocatedStackIssued
VIDEOFILES_EMPTY	InputParser_C.c, 209
videoFiles.h, 266	warplmage
VIDEOFILES END OF ITEMS	img_warp.c, 102
videoFiles.h, 266	img_warp.h, 103
VIDEOFILES FLV	wday
videoFiles.h, 266	timestamp, 39
VIDEOFILES H263	web_root_path
videoFiles.h, 266	AmmServer RH Context, 19
VIDEOFILES H264	webserver_root
videoFiles.h, 266	AmmServer_Instance, 16
VIDEOFILES MKV	helloworld.c, 43
videoFiles.h, 266	PreSpawnedThread, 38
VIDEOFILES MP4	ScriptRunner/main.cpp, 326
videoFiles.h, 266	Services/AmmarServer/main.c, 79
VIDEOFILES MPEG	Services/GeoPosShare/main.c, 82
videoFiles.h, 266	Services/MyLoader/main.c, 85
VIDEOFILES MPEG4	Services/MyURL/main.c, 95
videoFiles.h, 266	Services/ScriptRunner/main.cpp, 332
VIDEOFILES WEBM	Services/SimpleTemplate/main.c, 97
videoFiles.h, 266	width
varSocketTimeoutREAD_seconds	
server_configuration.c, 236	Image, 28 WriteJPEGFile
server_configuration.h, 245	jpgInput.c, 105
varSocketTimeoutWRITE seconds	jpglnput.h, 108
server_configuration.c, 236	WriteJPEGInternal
server_configuration.h, 245	ipgInput.c, 106
ver	WriteJPEGMemory
InputParser.cpp, 201	ipgInput.c, 106
verified local resource	,, ,
HTTPHeader, 26	jpgInput.h, 108
Version	WritePPM
InputParser, 32	imaging.c, 100
InputParser.cpp, 201	imaging.h, 100
·	YELLOW
version.h	logs.h, 315
RC_FILEVERSION, 320	-
videoFiles.h	year
VIDEOFILES_3GP, 266	timestamp, 39
VIDEOFILES_AVI, 266	
VIDEOFILES_EMPTY, 266	
VIDEOFILES_END_OF_ITEMS, 266	
VIDEOFILES_FLV, 266	
VIDEOFILES_H263, 266	
VIDEOFILES_H264, 266	
VIDEOFILES MKV. 266	