AmmarServer

Generated by Doxygen 1.8.6

Tue Sep 1 2015 04:01:23

Contents

1 AmmarServer 1
1.1 Introduction and History 1
1.2 What Is it ? 2
1.3 Coding Style 2
1.4 Future Plans 2
1.5 Deployment 3
1.6 Dependencies 3

2 Bug List 5

3 Data Structure Index 9
3.1 Data Structures 9

4 File Index 11
4.1 File List 11

5 Data Structure Documentation 15
5.1 _list_item Struct Reference 15
5.1.1 Field Documentation 15
5.1.1.1 curr 15
5.1.1.2 item 15
5.1.1.3 next 15
5.1.1.4 ptr 15
5.2 AmmServer_DynamicRequest Struct Reference 15
5.2.1 Detailed Description 16
5.2.2 Field Documentation 16
5.2.2.1 clientID 16
5.2.2.2 compressedContent 16
5.2.2.3 compressedContentSize 16
5.2.2.4 content 16
5.2.2.5 contentContainsPathToFileToBeStreamed 16
5.2.2.6 contentSize 16
5.2.2.7 GET_request 16

5.2.2.8 GET_request_length 16
5.2.2.9 headerResponse 16
5.2.2.10 MAXcompressedContentSize 16
5.2.2.11 MAXcontentSize 16
5.2.2.12 POST_request 16
5.2.2.13 POST_request_length 16
5.3 AmmServer_Instance Struct Reference 16
5.3.1 Detailed Description 17
5.3.2 Field Documentation 17
5.3.2.1 cache 17
5.3.2.2 cacheHashMap 17
5.3.2.3 cacheVersionETag 17
5.3.2.4 CLIENT_THREADS_STARTED 17
5.3.2.5 CLIENT_THREADS_STOPPED 17
5.3.2.6 clientList 17
5.3.2.7 clientRequestHandlerOverrideContext 17
5.3.2.8 files_open 18
5.3.2.9 instanceName 18
5.3.2.10 loaded_cache_items 18
5.3.2.11 loaded_cache_items_Kbytes 18
5.3.2.12 pause_server 18
5.3.2.13 prespawn_jobs_finished 18
5.3.2.14 prespawn_jobs_started 18
5.3.2.15 prespawn_turn_to_serve 18
5.3.2.16 prespawned_pool 18
5.3.2.17 server_running 18
5.3.2.18 server_thread_id 18
5.3.2.19 serversock 18
5.3.2.20 settings 18
5.3.2.21 stop_server 18
5.3.2.22 templates_root 18
5.3.2.23 threads_pool 18
5.3.2.24 webserver_root 18
5.4 AmmServer_Instance_Settings Struct Reference 18
5.4.1 Detailed Description 18
5.4.2 Field Documentation 19
5.4.2.1 BASE64PASSWORD 19
5.4.2.2 BINDING_PORT 19
5.4.2.3 PASSWORD 19
5.4.2.4 PASSWORD_PROTECTION 19

5.4.2.5 USERNAME 19
5.5 AmmServer_MemoryHandler Struct Reference 19
5.5.1 Detailed Description 19
5.5.2 Field Documentation 19
5.5.2.1 content 19
5.5.2.2 contentCurrentLength 19
5.5.2.3 contentSize 19
5.6 AmmServer_RequestOverride_Context Struct Reference 19
5.6.1 Detailed Description 20
5.6.2 Field Documentation 20
5.6.2.1 request 20
5.6.2.2 request_override_callback 20
5.6.2.3 requestHeader 20
5.7 AmmServer_RH_Context Struct Reference 20
5.7.1 Detailed Description 20
5.7.2 Field Documentation 21
5.7.2.1 callback_cooldown 21
5.7.2.2 callback_every_x_msec 21
5.7.2.3 dynamicRequestCallbackFunction 21
5.7.2.4 executedNow 21
5.7.2.5 last_callback 21
5.7.2.6 requestContext 21
5.7.2.7 resource_name 21
5.7.2.8 RH_Scenario 21
5.7.2.9 web_root_path 21
5.8 board Struct Reference 21
5.8.1 Field Documentation 21
5.8.1.1 active 21
5.8.1.2 currentThreads 21
5.8.1.3 currentUsers 21
5.8.1.4 hidden 21
5.8.1.5 imageUID 22
5.8.1.6 maxThreads 22
5.8.1.7 name 22
5.8.1.8 postUID 22
5.8.1.9 threadQueue 22
5.8.1.10 threads 22
5.8.1.11 threadUID 22
5.9 cache_item Struct Reference 22
5.9.1 Detailed Description 22

5.9.2 Field Documentation 22
5.9.2.1 compressedContent 22
5.9.2.2 compressedContentSize 22
5.9.2.3 content 22
5.9.2.4 contentSize 22
5.9.2.5 contentTypeID 22
5.9.2.6 doNOTCacheRule 23
5.9.2.7 dynamicRequest 23
5.9.2.8 dynamicRequestCallbackFunction 23
5.9.2.9 modification 23
5.10 clientListContext Struct Reference 23
5.10.1 Detailed Description 23
5.10.2 Field Documentation 23
5.10.2.1 userList 23
5.11 fastStringParser Struct Reference 23
5.11.1 Detailed Description 24
5.11.2 Field Documentation 24
5.11.2.1 contents 24
5.11.2.2 functionName 24
5.11.2.3 longestStringLength 24
5.11.2.4 MAXstringsLoaded 24
5.11.2.5 shortestStringLength 24
5.11.2.6 stringsLoaded 24
5.12 fspString Struct Reference 24
5.12.1 Detailed Description 24
5.12.2 Field Documentation 24
5.12.2.1 str 24
5.12.2.2 strIDFriendly 24
5.12.2.3 strLength 24
5.13 guard_byte Struct Reference 25
5.13.1 Field Documentation 25
5.13.1.1 checksum 25
5.14 hashMap Struct Reference 25
5.14.1 Detailed Description 25
5.14.2 Field Documentation 25
5.14.2.1 clearItemCallbackFunction 25
5.14.2.2 curNumberOfEntries 25
5.14.2.3 entries 25
5.14.2.4 entryAllocationStep 25
5.14.2.5 hm_addLock 25

5.14.2.6 hm_fileLock 25
5.14.2.7 maxNumberOfEntries 26
5.15 hashMapEntry Struct Reference 26
5.15.1 Detailed Description 26
5.15.2 Field Documentation 26
5.15.2.1 hits 26
5.15.2.2 key 26
5.15.2.3 keyHash 26
5.15.2.4 keyLength 26
5.15.2.5 payload 26
5.15.2.6 payloadLength 26
5.16 htmlContent Struct Reference 26
5.16.1 Field Documentation 27
5.16.1.1 currentDataLength 27
5.16.1.2 data 27
5.16.1.3 totalDataLength 27
5.17 HTTPHeader Struct Reference 27
5.17.1 Detailed Description 28
5.17.2 Field Documentation 28
5.17.2.1 authorized 28
5.17.2.2 boundary 28
5.17.2.3 boundaryLength 28
5.17.2.4 contentDisposition 28
5.17.2.5 contentDispositionLength 28
5.17.2.6 ContentLength 28
5.17.2.7 contentType 28
5.17.2.8 contentTypeLength 28
5.17.2.9 cookie 28
5.17.2.10 cookieLength 28
5.17.2.11 eTag 28
5.17.2.12 eTagLength 28
5.17.2.13 GETquery 28
5.17.2.14 headerRAW 28
5.17.2.15 headerRAWSize 28
5.17.2.16 host 28
5.17.2.17 hostLength 28
5.17.2.18 keepalive 28
5.17.2.19 POSTrequest 28
5.17.2.20 POSTrequestSize 28
5.17.2.21 range_end 28

5.17.2.22 range_start 28
5.17.2.23 referer 28
5.17.2.24 refererLength 28
5.17.2.25 requestType 29
5.17.2.26 resource 29
5.17.2.27 supports_compression 29
5.17.2.28 userAgent 29
5.17.2.29 userAgentLength 29
5.17.2.30 verified_local_resource 29
5.18 HTTPTransaction Struct Reference 29
5.18.1 Detailed Description 29
5.18.2 Field Documentation 29
5.18.2.1 clientListID 29
5.18.2.2 clientSock 29
5.18.2.3 incomingHeader 29
5.18.2.4 instance 29
5.18.2.5 outgoingBody 29
5.18.2.6 outgoingBodySize 29
5.18.2.7 prespawnedThreadFlag 30
5.18.2.8 resourceCacheID 30
5.18.2.9 threadID 30
5.19 Image Struct Reference 30
5.19.1 Field Documentation 30
5.19.1.1 depth 30
5.19.1.2 height 30
5.19.1.3 imageSize 30
5.19.1.4 pixels 30
5.19.1.5 width 30
5.20 image_region_type Struct Reference 30
5.20.1 Field Documentation 31
5.20.1.1 border 31
5.20.1.2 cmap 31
5.20.1.3 height 31
5.20.1.4 vis 31
5.20.1.5 visible_region 31
5.20.1.6 width 31
5.20.1.7 win 31
5.20.1.8 x_rootrel 31
5.20.1.9 x_vis 31
5.20.1.10 y_rootrel 31

5.20.1.11 y_vis 31
5.21 image_win_type Struct Reference 31
5.21.1 Field Documentation 31
5.21.1.1 border_width 31
5.21.1.2 cmap 31
5.21.1.3 height 31
5.21.1.4 parent 31
5.21.1.5 vis 32
5.21.1.6 width 32
5.21.1.7 win 32
5.21.1.8 x_rootrel 32
5.21.1.9 x_vis 32
5.21.1.10 y_rootrel 32
5.21.1.11 y_vis 32
5.22 InputParser Class Reference 32
5.22.1 Constructor & Destructor Documentation 32
5.22.1.1 InputParser 32
5.22.1.2 ∼InputParser 32
5.22.2 Member Function Documentation 33
5.22.2.1 DefaultDelimeterSetup 33
5.22.2.2 GetDelimeter 33
5.22.2.3 GetLowercaseWord 33
5.22.2.4 GetUpcaseWord 33
5.22.2.5 GetWord 33
5.22.2.6 GetWordChar 33
5.22.2.7 GetWordInt 33
5.22.2.8 GetWordLength 33
5.22.2.9 SeperateWords 33
5.22.2.10 SeperateWordsCC 33
5.22.2.11 SeperateWordsUC 33
5.22.2.12 SetDelimeter 33
5.22.2.13 Version 34
5.23 InputParserC Struct Reference 34
5.23.1 Field Documentation 34
5.23.1.1 container_end 34
5.23.1.2 container_start 34
5.23.1.3 cur_container_count 34
5.23.1.4 cur_delimeter_count 34
5.23.1.5 delimeters 34
5.23.1.6 guardbyte1 34

5.23.1.7 guardbyte2 34
5.23.1.8 guardbyte3 34
5.23.1.9 guardbyte4 34
5.23.1.10 local_allocation 35
5.23.1.11 max_container_count 35
5.23.1.12 max_delimeter_count 35
5.23.1.13 str 35
5.23.1.14 str_length 35
5.23.1.15 tokenlist 35
5.23.1.16 tokens_count 35
5.23.1.17 tokens_max 35
5.24 linkItemList Struct Reference 35
5.24.1 Field Documentation 35
5.24.1.1 currentItems 35
5.24.1.2 item 35
5.24.1.3 maxItems 35
5.25 linkLabelItem Struct Reference 35
5.25.1 Field Documentation 36
5.25.1.1 label 36
5.25.1.2 link 36
5.26 menuItemList Struct Reference 36
5.26.1 Field Documentation 36
5.26.1.1 currentItems 36
5.26.1.2 item 36
5.26.1.3 maxItems 36
5.27 my_XRegion Struct Reference 36
5.27.1 Field Documentation 36
5.27.1.1 extents 36
5.27.1.2 numRects 36
5.27.1.3 rects 37
5.27.1.4 size 37
5.28 myBox Struct Reference 37
5.28.1 Detailed Description 37
5.28.2 Field Documentation 37
5.28.2.1 x1 37
5.28.2.2 x2 37
5.28.2.3 y1 37
5.28.2.4 y2 37
5.29 OverlayInfo Struct Reference 38
5.29.1 Field Documentation 38

5.29.1.1 layer 38
5.29.1.2 pOverlayVisualInfo 38
5.29.1.3 transparentType 38
5.29.1.4 value 38
5.30 OverlayVisualPropertyRec Struct Reference 38
5.30.1 Detailed Description 38
5.30.2 Field Documentation 39
5.30.2.1 layer 39
5.30.2.2 transparentType 39
5.30.2.3 value 39
5.30.2.4 visualID 39
5.31 PassToHTTPThread Struct Reference 39
5.31.1 Detailed Description 39
5.31.2 Field Documentation 39
5.31.2.1 client 39
5.31.2.2 clientlen 39
5.31.2.3 clientsock 39
5.31.2.4 instance 40
5.31.2.5 ip 40
5.31.2.6 keep_var_on_stack 40
5.31.2.7 port 40
5.31.2.8 pre_spawned_thread 40
5.31.2.9 thread_id 40
5.32 PassToPreSpawnedThread Struct Reference 40
5.32.1 Field Documentation 40
5.32.1.1 i_adapt 40
5.32.1.2 instance 40
5.33 playlist Struct Reference 40
5.33.1 Field Documentation 41
5.33.1.1 item 41
5.33.1.2 maxItems 41
5.33.1.3 numberOfItems 41
5.33.1.4 playlistActiveItem 41
5.33.1.5 playlistState 41
5.34 playlistItem Struct Reference 41
5.34.1 Field Documentation 41
5.34.1.1 command 41
5.34.1.2 playFile 41
5.34.1.3 stopTime 41
5.34.1.4 triggerTime 41

5.35 post Struct Reference 41
5.35.1 Field Documentation 42
5.35.1.1 creation 42
5.35.1.2 fileCachedName 42
5.35.1.3 fileDimensionHeight 42
5.35.1.4 fileDimensionWidth 42
5.35.1.5 fileOriginalName 42
5.35.1.6 fileType 42
5.35.1.7 hasFile 42
5.35.1.8 message 42
5.35.1.9 messageSize 42
5.35.1.10 numberOfComplaints 42
5.35.1.11 op 42
5.35.1.12 password 42
5.36 postItem Struct Reference 42
5.36.1 Field Documentation 43
5.36.1.1 author 43
5.36.1.2 content 43
5.36.1.3 dateStr 43
5.36.1.4 tags 43
5.36.1.5 title 43
5.37 postItemList Struct Reference 43
5.37.1 Field Documentation 43
5.37.1.1 currentPosts 43
5.37.1.2 item 43
5.37.1.3 maxPosts 43
5.38 PreSpawnedThread Struct Reference 43
5.38.1 Detailed Description 44
5.38.2 Field Documentation 44
5.38.2.1 busy 44
5.38.2.2 client 44
5.38.2.3 clientlen 44
5.38.2.4 clientsock 44
5.38.2.5 instance 44
5.38.2.6 templates_root 44
5.38.2.7 thread_id 44
5.38.2.8 threadNum 44
5.38.2.9 webserver_root 44
5.39 site Struct Reference 44
5.39.1 Field Documentation 45

5.39.1.1 boards 45
5.39.1.2 maxNumberOfBoards 45
5.39.1.3 numberOfBoards 45
5.39.1.4 siteDescription 45
5.39.1.5 siteName 45
5.40 socialLinks Struct Reference 45
5.40.1 Field Documentation 45
5.40.1.1 facebookURL 45
5.40.1.2 twitterURL 45
5.40.1.3 youtubeURL 45
5.41 SQLiteSession Struct Reference 45
5.41.1 Field Documentation 46
5.41.1.1 db 46
5.41.1.2 err_msg 46
5.41.1.3 rc 46
5.41.1.4 res 46
5.42 tagItem Struct Reference 46
5.42.1 Field Documentation 46
5.42.1.1 tag 46
5.42.1.2 tagHash 46
5.43 tagItemList Struct Reference 46
5.43.1 Field Documentation 47
5.43.1.1 currentTags 47
5.43.1.2 item 47
5.43.1.3 maxTags 47
5.44 thread Struct Reference 47
5.44.1 Field Documentation 47
5.44.1.1 creation 47
5.44.1.2 lastReply 47
5.44.1.3 maxNumberOfReplies 47
5.44.1.4 numberOfImages 47
5.44.1.5 numberOfReplies 47
5.44.1.6 op 47
5.44.1.7 password 47
5.44.1.8 repliable 47
5.44.1.9 replies 47
5.44.1.10 sticky 47
5.44.1.11 title 48
5.45 time_snap Struct Reference 48
5.45.1 Field Documentation 48

5.45.1.1 difference 48
5.46 timestamp Struct Reference 48
5.46.1 Detailed Description 48
5.46.2 Field Documentation 49
5.46.2.1 day 49
5.46.2.2 day 49
5.46.2.3 hour 49
5.46.2.4 hour 49
5.46.2.5 minute 49
5.46.2.6 minute 49
5.46.2.7 month 49
5.46.2.8 month 49
5.46.2.9 second 49
5.46.2.10 second 49
5.46.2.11 wday 49
5.46.2.12 year 49
5.47 tokens Struct Reference 49
5.47.1 Field Documentation 49
5.47.1.1 length 49
5.47.1.2 token_start 49
5.48 URLDB Struct Reference 50
5.48.1 Field Documentation 50
5.48.1.1 longURL 50
5.48.1.2 shortURL 50
5.48.1.3 shortURLHash 50
5.49 UserAccountAuthenticationToken Struct Reference 50
5.49.1 Field Documentation 50
5.49.1.1 dummy 50
5.50 UserAccountDatabase Struct Reference 50
5.50.1 Field Documentation 51
5.50.1.1 dummy 51
5.50.1.2 lastAuthenticationToken 51
5.51 videoCollection Struct Reference 51
5.51.1 Field Documentation 51
5.51.1.1 MAX_numberOfVideos 51
5.51.1.2 numberOfLoadedVideos 51
5.51.1.3 video 51
5.52 videoItem Struct Reference 51
5.52.1 Field Documentation 52
5.52.1.1 comment 52

5.52.1.2 dislikes 52
5.52.1.3 filename 52
5.52.1.4 hashID 52
5.52.1.5 likes 52
5.52.1.6 tagsStr 52
5.52.1.7 thumbnail 52
5.52.1.8 title 52
5.52.1.9 views 52
5.52.1.10 visibility 52
5.53 website Struct Reference 52
5.53.1 Field Documentation 52
5.53.1.1 allowComments 52
5.53.1.2 allowPing 52
5.53.1.3 blogTitle 53
5.53.1.4 linksLeft 53
5.53.1.5 linksRight 53
5.53.1.6 menu 53
5.53.1.7 post 53
5.53.1.8 siteDescription 53
5.53.1.9 siteName 53
5.53.1.10 siteURL 53
5.53.1.11 social 53
5.53.1.12 widget 53
5.54 widgetItem Struct Reference 53
5.54.1 Field Documentation 53
5.54.1.1 content 53
5.54.1.2 label 53
5.54.1.3 link 53
5.55 widgetItemList Struct Reference 53
5.55.1 Field Documentation 54
5.55.1.1 currentItems 54
5.55.1.2 item 54
5.55.1.3 maxItems 54

6 File Documentation 55

6.1 doc/DoxygenMainpage.h File Reference 55
6.2 doc/helloworld.c File Reference 55
6.2.1 Function Documentation 55
6.2.1.1 close_dynamic_content 55
6.2.1.2 init_dynamic_content 55

6.2.1.3 main 56
6.2.1.4 prepare_helloworld_content_callback 56
6.2.2 Variable Documentation 56
6.2.2.1 helloworld 56
6.2.2.2 helloworld_times_shown 56
6.2.2.3 templates_root 56
6.2.2.4 webserver_root 56
6.3 src/AmmCaptcha/AmmCaptcha.h File Reference 56
6.3.1 Function Documentation 56
6.3.1.1 AmmCaptcha_destroy 56
6.3.1.2 AmmCaptcha_getCaptchaFrame 56
6.3.1.3 AmmCaptcha_getJPEGFileFromPixels 56
6.3.1.4 AmmCaptcha_initialize 56
6.3.1.5 AmmCaptcha_isReplyCorrect 57
6.3.1.6 testAmmCaptcha 57
6.4 src/AmmCaptcha/AmmCaptchaTester/main.c File Reference 57
6.4.1 Function Documentation 57
6.4.1.1 main 57
6.5 src/AmmCaptcha/main.c File Reference 57
6.5.1 Macro Definition Documentation 58
6.5.1.1 RANDOMIZE_AFTER_FAILED_ATTEMPT 58
6.5.2 Function Documentation 58
6.5.2.1 AmmCaptcha_copyCaptchaJPEGImageWithCopy 58
6.5.2.2 AmmCaptcha_destroy 58
6.5.2.3 AmmCaptcha_getCaptchaFrame 58
6.5.2.4 AmmCaptcha_getJPEGFileFromPixels 58
6.5.2.5 AmmCaptcha_initialize 58
6.5.2.6 AmmCaptcha_isReplyCorrect 58
6.5.2.7 AmmCaptcha_loadDictionary 58
6.5.2.8 convertExternalIDToInternal 59
6.5.2.9 RenderString 59
6.5.2.10 testAmmCaptcha 59
6.5.3 Variable Documentation 59
6.5.3.1 captchaStrings 59
6.5.3.2 fontRAW 59
6.5.3.3 fontX 59
6.5.3.4 fontY 59
6.6 src/AmmServerlib/InputParser/InputParser_C_Tester/main.c File Reference 59
6.6.1 Macro Definition Documentation 60
6.6.1.1 BLACK 60

6.6.1.2 BLUE 60
6.6.1.3 CYAN 60
6.6.1.4 GREEN 60
6.6.1.5 MAGENTA 60
6.6.1.6 max_ret_word 60
6.6.1.7 NORMAL 60
6.6.1.8 RED 60
6.6.1.9 WHITE 60
6.6.1.10 YELLOW 60
6.6.2 Function Documentation 60
6.6.2.1 IntermediateTests 60
6.6.2.2 main 60
6.6.2.3 ParseString 60
6.7 src/AmmServerlib/main.c File Reference 60
6.7.1 Function Documentation 64
6.7.1.1 _FILES 64
6.7.1.2 _GET 64
6.7.1.3 _POST 64
6.7.1.4 AmmServer_AddRequestHandler 64
6.7.1.5 AmmServer_AddResourceHandler 64
6.7.1.6 AmmServer_AllocateMemoryHandler 65
6.7.1.7 AmmServer_CheckIfHeaderBinaryAreTheSame 65
6.7.1.8 AmmServer_ConvertBufferToMemoryHandler 65
6.7.1.9 AmmServer_CopyMemoryHandler 65
6.7.1.10 AmmServer_CopyOverlappingDataContent 65
6.7.1.11 AmmServer_DirectoryExists 66
6.7.1.12 AmmServer_DoNOTCacheResource 66
6.7.1.13 AmmServer_DoNOTCacheResourceHandler 66
6.7.1.14 AmmServer_DynamicRequestReturnFile 66
6.7.1.15 AmmServer_EraseFile 67
6.7.1.16 AmmServer_Error 67
6.7.1.17 AmmServer_ExecuteCommandLine 67
6.7.1.18 AmmServer_ExecuteCommandLineNum 67
6.7.1.19 AmmServer_FileExists 68
6.7.1.20 AmmServer_FileIsVideo 68
6.7.1.21 AmmServer_FILES 68
6.7.1.22 AmmServer_FreeMemoryHandler 69
6.7.1.23 AmmServer_GeneralPrint 69
6.7.1.24 AmmServer_GETArg 69
6.7.1.25 AmmServer_GetInfo 69

6.7.1.26 AmmServer_GetIntSettingValue 69
6.7.1.27 AmmServer_GetStrSettingValue 70
6.7.1.28 AmmServer_GlobalTerminationHandler 71
6.7.1.29 AmmServer_InjectDataToBuffer 71
6.7.1.30 AmmServer_POSTArg 71
6.7.1.31 AmmServer_PreCacheFile 71
6.7.1.32 AmmServer_ReadFileToMemory 71
6.7.1.33 AmmServer_ReadFileToMemoryHandler 72
6.7.1.34 AmmServer_RegisterTerminationSignal 72
6.7.1.35 AmmServer_RemoveResourceHandler 72
6.7.1.36 AmmServer_ReplaceAllVarsInMemoryFile 73
6.7.1.37 AmmServer_ReplaceAllVarsInMemoryHandler 74
6.7.1.38 AmmServer_ReplaceCharInString 74
6.7.1.39 AmmServer_ReplaceVarInMemoryFile 74
6.7.1.40 AmmServer_ReplaceVarInMemoryHandler 74
6.7.1.41 AmmServer_Running 75
6.7.1.42 AmmServer_SaveDynamicRequest 75
6.7.1.43 AmmServer_SelfCheck 75
6.7.1.44 AmmServer_SetIntSettingValue 75
6.7.1.45 AmmServer_SetStrSettingValue 76
6.7.1.46 AmmServer_SignalCountAsBadClientBehaviour 76
6.7.1.47 AmmServer_Start 76
6.7.1.48 AmmServer_StartAdminInstance 76
6.7.1.49 AmmServer_StartWithArgs 77
6.7.1.50 AmmServer_Stop 77
6.7.1.51 AmmServer_StringIsHTMLSafe 77
6.7.1.52 AmmServer_Success 78
6.7.1.53 AmmServer_Version 78
6.7.1.54 AmmServer_Warning 78
6.7.1.55 AmmServer_WriteFileFromMemory 78
6.7.2 Variable Documentation 78
6.7.2.1 TerminationCallback 78
6.8 src/ScriptRunner/main.c File Reference 78
6.8.1 Macro Definition Documentation 80
6.8.1.1 ADMIN_BINDING_PORT 80
6.8.1.2 DEFAULT_BINDING_PORT 80
6.8.1.3 ENABLE_ADMIN_PAGE 80
6.8.1.4 ENABLE_CHAT_BOX 80
6.8.1.5 ENABLE_PASSWORD_PROTECTION 80
6.8.1.6 MAX_BINDING_PORT 80

6.8.1.7 MAX_COMMAND_SIZE 80
6.8.2 Function Documentation 80
6.8.2.1 close_dynamic_content 80
6.8.2.2 execute 80
6.8.2.3 getBackCommandLine 80
6.8.2.4 init_dynamic_content 80
6.8.2.5 joystickExecute 80
6.8.2.6 main 80
6.8.2.7 prepare_base_image 80
6.8.2.8 prepare_form_content_callback 80
6.8.2.9 prepare_index_content_callback 81
6.8.2.10 prepare_stats_content_callback 81
6.8.2.11 prepare_top_image 81
6.8.2.12 replaceChar 81
6.8.2.13 store_new_configuration_callback 81
6.8.2.14 termination_handler 81
6.8.3 Variable Documentation 81
6.8.3.1 admin_root 81
6.8.3.2 admin_server 81
6.8.3.3 base_image 81
6.8.3.4 chatbox 81
6.8.3.5 default_server 81
6.8.3.6 form 81
6.8.3.7 GET_override 81
6.8.3.8 indexPage 81
6.8.3.9 page 81
6.8.3.10 pageLength 81
6.8.3.11 random_chars 81
6.8.3.12 settings 81
6.8.3.13 stats 81
6.8.3.14 templates_root 81
6.8.3.15 top_image 82
6.8.3.16 webserver_root 82
6.9 src/Services/AmmarServer/main.c File Reference 82
6.9.1 Macro Definition Documentation 83
6.9.1.1 ADMIN_BINDING_PORT 83
6.9.1.2 DEFAULT_BINDING_PORT 83
6.9.1.3 ENABLE_ADMIN_PAGE 83
6.9.1.4 ENABLE_CHAT_BOX 83
6.9.1.5 ENABLE_GET_DEBUGGING 83

6.9.1.6 ENABLE_PASSWORD_PROTECTION 83
6.9.1.7 ENABLE_STOP_PAGE 83
6.9.1.8 logEcho 83
6.9.1.9 MAX_BINDING_PORT 83
6.9.1.10 MAX_SCRIPT_RESPONSE_SIZE 83
6.9.1.11 WEBSERVERROOT 83
6.9.2 Function Documentation 83
6.9.2.1 close_dynamic_content 83
6.9.2.2 debug_get_callback 83
6.9.2.3 executeScriptFunction 83
6.9.2.4 init_dynamic_content 83
6.9.2.5 main 83
6.9.2.6 prepare_chatbox_content_callback 84
6.9.2.7 prepare_form_content_callback 84
6.9.2.8 prepare_gps_content_callback 84
6.9.2.9 prepare_random_content_callback 84
6.9.2.10 prepare_stats_content_callback 84
6.9.2.11 request_override_callback 84
6.9.2.12 stop_callback 84
6.9.3 Variable Documentation 84
6.9.3.1 admin_root 84
6.9.3.2 admin_server 84
6.9.3.3 chatbox 84
6.9.3.4 default_server 84
6.9.3.5 executeScript 84
6.9.3.6 executeScriptRC 84
6.9.3.7 form 84
6.9.3.8 fresh 84
6.9.3.9 GET_override 84
6.9.3.10 getdbg 84
6.9.3.11 gps 84
6.9.3.12 random_chars 84
6.9.3.13 stats 84
6.9.3.14 stop 85
6.9.3.15 templates_root 85
6.9.3.16 webserver_root 85
6.10 src/Services/CinemaPilot/main.c File Reference 85
6.10.1 Macro Definition Documentation 86
6.10.1.1 DEFAULT_BINDING_PORT 86
6.10.2 Enumeration Type Documentation 86

6.10.2.1 commandType 86
6.10.2.2 stateType 86
6.10.3 Function Documentation 87
6.10.3.1 close_dynamic_content 87
6.10.3.2 executePlaylist 87
6.10.3.3 executePlaylistCurrentItem 87
6.10.3.4 init_dynamic_content 87
6.10.3.5 intermission 87
6.10.3.6 issueCommandToMplayer 87
6.10.3.7 keepalivePlaylist 87
6.10.3.8 main 87
6.10.3.9 pauseMplayer 87
6.10.3.10 prepare_indexPage 87
6.10.3.11 prepare_random_content_callback 87
6.10.3.12 prepare_remoteControl_callback 87
6.10.3.13 prepare_stats_content_callback 87
6.10.3.14 processCommand 87
6.10.3.15 readPlaylist 87
6.10.3.16 request_override_callback 88
6.10.3.17 resumeMplayer 88
6.10.3.18 startMplayer 88
6.10.3.19 stopMplayer 88
6.10.4 Variable Documentation 88
6.10.4.1 default_server 88
6.10.4.2 fullScreenViewerPath 88
6.10.4.3 GET_override 88
6.10.4.4 indexPage 88
6.10.4.5 movieList 88
6.10.4.6 mplayerControllerPath 88
6.10.4.7 random_chars 88
6.10.4.8 remoteControl 88
6.10.4.9 stats 88
6.10.4.10 templates_root 88
6.10.4.11 webserver_root 88
6.11 src/Services/GeoPosShare/main.c File Reference 88
6.11.1 Macro Definition Documentation 89
6.11.1.1 ADMIN_BINDING_PORT 89
6.11.1.2 DEFAULT_BINDING_PORT 89
6.11.1.3 ENABLE_ADMIN_PAGE 89
6.11.1.4 EPOCH_YEAR_IN_TM_YEAR 89

6.11.1.5 MAX_BINDING_PORT 90
6.11.2 Function Documentation 90
6.11.2.1 appendGPS_OSM_Format 90
6.11.2.2 appendGPSMessage 90
6.11.2.3 close_dynamic_content 90
6.11.2.4 init_dynamic_content 90
6.11.2.5 main 90
6.11.2.6 prepare_apk_link 90
6.11.2.7 prepare_gps_content_callback 90
6.11.2.8 prepare_indexPage 90
6.11.2.9 prepare_interestPoints 90
6.11.2.10 request_override_callback 90
6.11.3 Variable Documentation 90
6.11.3.1 admin_root 90
6.11.3.2 android 90
6.11.3.3 apk 90
6.11.3.4 default_server 90
6.11.3.5 GET_override 91
6.11.3.6 gps 91
6.11.3.7 indexPage 91
6.11.3.8 interestPoints 91
6.11.3.9 templates_root 91
6.11.3.10 webserver_root 91
6.12 src/Services/HabChan/main.c File Reference 91
6.12.1 Macro Definition Documentation 92
6.12.1.1 ADMIN_BINDING_PORT 92
6.12.1.2 DEFAULT_BINDING_PORT 92
6.12.1.3 MAX_BINDING_PORT 92
6.12.1.4 MAX_SCRIPT_RESPONSE_SIZE 92
6.12.1.5 WEBSERVERROOT 92
6.12.2 Function Documentation 92
6.12.2.1 close_dynamic_content 92
6.12.2.2 init_dynamic_content 92
6.12.2.3 main 92
6.12.3 Variable Documentation 92
6.12.3.1 boardIndexView 92
6.12.3.2 postReceiver 92
6.12.3.3 templates_root 92
6.12.3.4 threadIndexView 92
6.12.3.5 threadView 92

6.12.3.6 webserver_root 92
6.13 src/Services/MyBlog/main.c File Reference 92
6.13.1 Macro Definition Documentation 93
6.13.1.1 DEFAULT_BINDING_PORT 93
6.13.1.2 TEST_INDEX_GENERATION_ONLY 93
6.13.2 Function Documentation 93
6.13.2.1 close_dynamic_content 93
6.13.2.2 init_dynamic_content 93
6.13.2.3 main 93
6.13.2.4 prepare_random_content_callback 93
6.13.2.5 request_override_callback 93
6.13.3 Variable Documentation 93
6.13.3.1 default_server 93
6.13.3.2 GET_override 94
6.13.3.3 random_chars 94
6.13.3.4 stats 94
6.13.3.5 templates_root 94
6.13.3.6 webserver_root 94
6.14 src/Services/MyLoader/main.c File Reference 94
6.14.1 Macro Definition Documentation 94
6.14.1.1 DEFAULT_BINDING_PORT 94
6.14.2 Function Documentation 94
6.14.2.1 close_dynamic_content 94
6.14.2.2 init_dynamic_content 95
6.14.2.3 main 95
6.14.2.4 prepare_stats_content_callback 95
6.14.2.5 processUploadCallback 95
6.14.2.6 request_override_callback 95
6.14.3 Variable Documentation 95
6.14.3.1 default_server 95
6.14.3.2 GET_override 95
6.14.3.3 stats 95
6.14.3.4 templates_root 95
6.14.3.5 uploadProcessor 95
6.14.3.6 webserver_root 95
6.15 src/Services/MyRemoteDesktop/main.c File Reference 95
6.15.1 Macro Definition Documentation 96
6.15.1.1 ALLOW_REMOTE_CONTROL 96
6.15.1.2 DEFAULT_BINDING_PORT 96
6.15.1.3 XWDLIB_BRIDGE 96

6.15.2 Function Documentation 96
6.15.2.1 close_dynamic_content 96
6.15.2.2 init_dynamic_content 96
6.15.2.3 main 96
6.15.2.4 prepare_command_content_callback 96
6.15.2.5 prepare_index_content_callback 97
6.15.2.6 prepare_screen_content_callback 97
6.15.3 Variable Documentation 97
6.15.3.1 commandContext 97
6.15.3.2 default_server 97
6.15.3.3 GET_override 97
6.15.3.4 indexPage 97
6.15.3.5 indexPageContext 97
6.15.3.6 indexPageLength 97
6.15.3.7 indexPagePath 97
6.15.3.8 screenContext 97
6.15.3.9 templates_root 97
6.15.3.10 webserver_root 97
6.16 src/Services/MyRemoteDesktop/xwd-1.0.5/main.c File Reference 97
6.16.1 Macro Definition Documentation 98
6.16.1.1 FEEP_VOLUME 98
6.16.1.2 lowbit 98
6.16.2 Typedef Documentation 98
6.16.2.1 Pixel 98
6.16.3 Function Documentation 98
6.16.3.1 _swaplong 98
6.16.3.2 _swapshort 98
6.16.3.3 closeXwdLib 98
6.16.3.4 Get_XColors 98
6.16.3.5 getScreen 98
6.16.3.6 Image_Size 98
6.16.3.7 initXwdLib 98
6.16.3.8 main 98
6.16.3.9 usage 99
6.16.3.10 Window_Dump 99
6.16.4 Variable Documentation 99
6.16.4.1 frame_only 99
6.16.4.2 i 99
6.16.4.3 out_file 99
6.16.4.4 target_win 99

6.17 src/Services/MyTube/main.c File Reference 99
6.17.1 Macro Definition Documentation 100
6.17.1.1 DEFAULT_BINDING_PORT 100
6.17.1.2 DO_DYNAMIC_THUMBNAILS 100
6.17.1.3 UPDATE_ALL_THUMBNAILS_ON_LAUNCH 100
6.17.1.4 VIDEO_FILES_PATH_1 100
6.17.1.5 VIDEO_FILES_PATH_2 100
6.17.1.6 VIDEO_FILES_PATH_3 100
6.17.2 Function Documentation 100
6.17.2.1 close_dynamic_content 100
6.17.2.2 init_dynamic_content 100
6.17.2.3 main 100
6.17.2.4 serve_favicon 101
6.17.2.5 serve_index 101
6.17.2.6 serve_interact 101
6.17.2.7 serve_random_videopage 101
6.17.2.8 serve_thumbnail 101
6.17.2.9 serve_videofile 101
6.17.2.10 serve_videopage 101
6.17.2.11 thumbnailAllVideoDatabase 101
6.17.3 Variable Documentation 101
6.17.3.1 database_root 101
6.17.3.2 default_server 101
6.17.3.3 favicon 101
6.17.3.4 faviconContext 101
6.17.3.5 GET_override 101
6.17.3.6 indexContext 101
6.17.3.7 indexPage 101
6.17.3.8 interactContext 101
6.17.3.9 myTube 101
6.17.3.10 random_chars 101
6.17.3.11 randomVideoFileContext 101
6.17.3.12 templates_root 102
6.17.3.13 thumbnailContext 102
6.17.3.14 video_root 102
6.17.3.15 videoFileContext 102
6.17.3.16 videoPageContext 102
6.17.3.17 webserver_root 102
6.18 src/Services/MyURL/main.c File Reference 102
6.18.1 Macro Definition Documentation 103

6.18.1.1 DEFAULT_BINDING_PORT 103
6.18.1.2 DYNAMIC_PAGES_MEMORY_COMMITED 103
6.18.1.3 ENABLE_CAPTCHA_SYSTEM 103
6.18.1.4 LINK_ALLOCATION_STEP 103
6.18.1.5 MAX_BINDING_PORT 104
6.18.1.6 MAX_CAPTCHA_JPG_SIZE 104
6.18.1.7 MAX_LINKS 104
6.18.1.8 MAX_LONG_URL_SIZE 104
6.18.1.9 MAX_TO_SIZE 104
6.18.1.10 REGROUP_AFTER_X_UNSORTED_LINKS 104
6.18.1.11 USE_BINARY_SEARCH 104
6.18.2 Function Documentation 104
6.18.2.1 Add_MyURL 104
6.18.2.2 allocateLinksIfNeeded 104
6.18.2.3 Append2MyURLDBFile 104
6.18.2.4 close_dynamic_content 104
6.18.2.5 Find_longURL 104
6.18.2.6 Find_longURLSerial 104
6.18.2.7 Get_longURL 104
6.18.2.8 hashURL 104
6.18.2.9 init_dynamic_content 104
6.18.2.10 is_an_unsafe_str 104
6.18.2.11 isURLDBSorted 104
6.18.2.12 LoadMyURLDBFile 105
6.18.2.13 main 105
6.18.2.14 printURLDB 105
6.18.2.15 resolveRequest 105
6.18.2.16 ResortDB 105
6.18.2.17 ReWriteMyURLDBFile 105
6.18.2.18 serve_captcha_page 105
6.18.2.19 serve_create_url_page 105
6.18.2.20 serve_error_url_page 105
6.18.2.21 serve_goto_url_page 105
6.18.2.22 struct_cmp_urldb_items 105
6.18.3 Variable Documentation 105
6.18.3.1 allocated_links 105
6.18.3.2 captcha_url 105
6.18.3.3 create_url 105
6.18.3.4 db_addIDLock 105
6.18.3.5 db_file 105

6.18.3.6 db_fileLock 106
6.18.3.7 default_failed 106
6.18.3.8 error_url 106
6.18.3.9 goto_url 106
6.18.3.10 indexPage 106
6.18.3.11 indexPageLength 106
6.18.3.12 indexPagePath 106
6.18.3.13 links 106
6.18.3.14 loaded_links 106
6.18.3.15 myurl_server 106
6.18.3.16 requestResolver 106
6.18.3.17 service_filename 106
6.18.3.18 service_filename_noslash 106
6.18.3.19 service_root 106
6.18.3.20 service_root_withoutfilename 106
6.18.3.21 sorted_links 106
6.18.3.22 templates_root 106
6.18.3.23 webserver_root 106
6.19 src/Services/SimpleTemplate/main.c File Reference 106
6.19.1 Macro Definition Documentation 107
6.19.1.1 DEFAULT_BINDING_PORT 107
6.19.2 Function Documentation 107
6.19.2.1 close_dynamic_content 107
6.19.2.2 init_dynamic_content 107
6.19.2.3 main 107
6.19.2.4 prepare_random_content_callback 107
6.19.2.5 prepare_stats_content_callback 107
6.19.2.6 request_override_callback 107
6.19.3 Variable Documentation 107
6.19.3.1 default_server 107
6.19.3.2 GET_override 107
6.19.3.3 random_chars 107
6.19.3.4 stats 107
6.19.3.5 templates_root 108
6.19.3.6 webserver_root 108
6.20 src/Services/SQLiteServer/main.c File Reference 108
6.20.1 Macro Definition Documentation 108
6.20.1.1 DEFAULT_BINDING_PORT 108
6.20.2 Function Documentation 108
6.20.2.1 close_dynamic_content 108

6.20.2.2 init_dynamic_content 108
6.20.2.3 main 109
6.20.2.4 prepare_cars_content_callback 109
6.20.2.5 prepare_stats_content_callback 109
6.20.2.6 request_override_callback 109
6.20.3 Variable Documentation 109
6.20.3.1 default_server 109
6.20.3.2 GET_override 109
6.20.3.3 random_chars 109
6.20.3.4 sqliteSession 109
6.20.3.5 stats 109
6.20.3.6 templates_root 109
6.20.3.7 webserver_root 109
6.21 src/StringRecognizer/main.c File Reference 109
6.21.1 Function Documentation 109
6.21.1.1 main 109
6.22 src/UserAccounts/main.c File Reference 110
6.22.1 Function Documentation 110
6.22.1.1 uadb_authenticateUser 110
6.22.1.2 uadb_closeUserAccountDatabase 110
6.22.1.3 uadb_initializeUserAccountDatabase 110
6.22.1.4 uadb_loginUser 110
6.23 src/AmmCaptcha/imaging.c File Reference 110
6.23.1 Macro Definition Documentation 111
6.23.1.1 DISPLAY_DEBUG_INFO 111
6.23.1.2 PPMREADBUFLEN 111
6.23.1.3 READ_CREATES_A_NEW_PIXEL_BUFFER 111
6.23.2 Function Documentation 111
6.23.2.1 bitBltImage 111
6.23.2.2 bitBltImageRotated 111
6.23.2.3 copyImage 111
6.23.2.4 createImage 111
6.23.2.5 destroyImage 111
6.23.2.6 ReadPPM 111
6.23.2.7 WritePPM 111
6.24 src/AmmCaptcha/imaging.h File Reference 111
6.24.1 Function Documentation 112
6.24.1.1 bitBltImage 112
6.24.1.2 copyImage 112
6.24.1.3 createImage 112

6.24.1.4 destroyImage 112
6.24.1.5 ReadPPM 112
6.24.1.6 WritePPM 112
6.25 src/AmmCaptcha/img_warp.c File Reference 112
6.25.1 Macro Definition Documentation 112
6.25.1.1 ABS 112
6.25.1.2 ABSDIFF 112
6.25.2 Function Documentation 112
6.25.2.1 coolPHPWave 112
6.25.2.2 warpImage 113
6.26 src/AmmCaptcha/img_warp.h File Reference 113
6.26.1 Function Documentation 113
6.26.1.1 coolPHPWave 113
6.26.1.2 warpImage 113
6.27 src/AmmCaptcha/jpgInput.c File Reference 113
6.27.1 Function Documentation 114
6.27.1.1 empty_buffer 114
6.27.1.2 fastJPGHeaderCheck 114
6.27.1.3 init_buffer 114
6.27.1.4 jpegtest 114
6.27.1.5 ReadJPEG 114
6.27.1.6 term_buffer 114
6.27.1.7 WriteJPEGFile 114
6.27.1.8 WriteJPEGInternal 114
6.27.1.9 WriteJPEGMemory 115
6.28 src/AmmCaptcha/jpgInput.h File Reference 115
6.28.1 Macro Definition Documentation 115
6.28.1.1 USE_JPG_FILES 115
6.28.2 Function Documentation 115
6.28.2.1 ReadJPEG 115
6.28.2.2 WriteJPEGFile 115
6.28.2.3 WriteJPEGMemory 115
6.29 src/AmmServerlib/AmmServerlib.h File Reference 115
6.29.1 Detailed Description 120
6.29.2 Macro Definition Documentation 120
6.29.2.1 AMMAR_SERVER_HTTP_HEADER_SPEC 120
6.29.2.2 MAX_FILE_PATH 120
6.29.2.3 MAX_INSTANCE_NAME_STRING 120
6.29.2.4 MAX_IP_STRING_SIZE 120
6.29.2.5 MAX_QUERY 120

6.29.2.6 MAX_RESOURCE . . . 120
6.29.2.7 POPEN_BUFFER_SIZE . . . 120
6.29.3 Enumeration Type Documentation . . . 120
6.29.3.1 AmmServInfos . . . 120
6.29.3.2 AmmServSettings . . . 120
6.29.3.3 AmmServStrSettings . . . 121
6.29.3.4 RHScenarios . . . 121
6.29.3.5 TypesOfRequests . . . 121
6.29.4 Function Documentation . . . 121
6.29.4.1 _FILES . . . 121
6.29.4.2 _GET . . . 122
6.29.4.3 _POST . . . 122
6.29.4.4 AmmServer_AddRequestHandler . . . 122
6.29.4.5 AmmServer_AddResourceHandler . . . 122
6.29.4.6 AmmServer_AllocateMemoryHandler . . . 123
6.29.4.7 AmmServer_CheckIfHeaderBinaryAreTheSame . . . 123
6.29.4.8 AmmServer_CopyMemoryHandler . . . 123
6.29.4.9 AmmServer_CopyOverlappingDataContent . . . 123
6.29.4.10 AmmServer_DirectoryExists . . . 123
6.29.4.11 AmmServer_DoNOTCacheResource . . . 124
6.29.4.12 AmmServer_DoNOTCacheResourceHandler . . . 124
6.29.4.13 AmmServer_DynamicRequestReturnFile . . . 124
6.29.4.14 AmmServer_EraseFile . . . 124
6.29.4.15 AmmServer_Error . . . 125
6.29.4.16 AmmServer_ExecuteCommandLine . . . 125
6.29.4.17 AmmServer_ExecuteCommandLineNum . . . 125
6.29.4.18 AmmServer_FileExists . . . 126
6.29.4.19 AmmServer_FileIsVideo . . . 126
6.29.4.20 AmmServer_FILES . . . 126
6.29.4.21 AmmServer_FreeMemoryHandler . . . 126
6.29.4.22 AmmServer_GETArg . . . 126
6.29.4.23 AmmServer_GetInfo . . . 127
6.29.4.24 AmmServer_GetIntSettingValue . . . 127
6.29.4.25 AmmServer_GetStrSettingValue . . . 127
6.29.4.26 AmmServer_InjectDataToBuffer . . . 128
6.29.4.27 AmmServer_POSTArg . . . 129
6.29.4.28 AmmServer_ReadFileToMemory . . . 129
6.29.4.29 AmmServer_ReadFileToMemoryHandler . . . 129
6.29.4.30 AmmServer_RegisterTerminationSignal . . . 130
6.29.4.31 AmmServer_RemoveResourceHandler . . . 131

6.29.4.32 AmmServer_ReplaceAllVarsInMemoryFile 131
6.29.4.33 AmmServer_ReplaceAllVarsInMemoryHandler 131
6.29.4.34 AmmServer_ReplaceCharInString 131
6.29.4.35 AmmServer_ReplaceVarInMemoryFile 132
6.29.4.36 AmmServer_ReplaceVarInMemoryHandler 132
6.29.4.37 AmmServer_Running 132
6.29.4.38 AmmServer_SaveDynamicRequest 132
6.29.4.39 AmmServer_SelfCheck 133
6.29.4.40 AmmServer_SetIntSettingValue 133
6.29.4.41 AmmServer_SetStrSettingValue 133
6.29.4.42 AmmServer_SignalCountAsBadClientBehaviour 134
6.29.4.43 AmmServer_Start 134
6.29.4.44 AmmServer_StartAdminInstance 134
6.29.4.45 AmmServer_StartWithArgs 134
6.29.4.46 AmmServer_Stop 135
6.29.4.47 AmmServer_StringIsHTMLSafe 135
6.29.4.48 AmmServer_Success 135
6.29.4.49 AmmServer_Version 135
6.29.4.50 AmmServer_Warning 136
6.29.4.51 AmmServer_WriteFileFromMemory 136
6.30 src/AmmServerlib/AString/AString.c File Reference 136
6.30.1 Macro Definition Documentation 137
6.30.1.1 BLACK 137
6.30.1.2 GREEN 137
6.30.1.3 NORMAL 137
6.30.1.4 RED 137
6.30.1.5 YELLOW 137
6.30.2 Function Documentation 137
6.30.2.1 astringCopyOverlappingDataContent 137
6.30.2.2 astringInjectDataToBuffer 137
6.30.2.3 astringInjectDataToMemoryHandler 137
6.30.2.4 astringReadFileToMemory 137
6.30.2.5 astringReplaceAllInstancesOfVarInMemoryFile 137
6.30.2.6 astringReplaceVarInMemoryFile 138
6.30.2.7 astringWriteFileFromMemory 138
6.30.2.8 myStupidMemcpy 138
6.31 src/AmmServerlib/AString/AString.h File Reference 138
6.31.1 Detailed Description 138
6.31.2 Function Documentation 138
6.31.2.1 astringCopyOverlappingDataContent 138

6.31.2.2 astringInjectDataToBuffer 138
6.31.2.3 astringInjectDataToMemoryHandler 138
6.31.2.4 astringReadFileToMemory 139
6.31.2.5 astringReplaceAllInstancesOfVarInMemoryFile 139
6.31.2.6 astringReplaceVarInMemoryFile 139
6.31.2.7 astringWriteFileFromMemory 139
6.32 src/AmmServerlib/cache/client_list.c File Reference 139
6.32.1 Macro Definition Documentation 140
6.32.1.1 COMPILE_WITH_CLIENT_LIST 140
6.32.2 Function Documentation 140
6.32.2.1 clientList_close 140
6.32.2.2 clientList_GetClientId 140
6.32.2.3 clientList_initialize 140
6.32.2.4 clientList_isClientAllowedToMakeAConnection 140
6.32.2.5 clientList_isClientAllowedToUseResource 140
6.32.2.6 clientList_isClientBanned 141
6.32.2.7 clientList_signalClientStoppedUsingResource 141
6.33 src/AmmServerlib/cache/client_list.h File Reference 141
6.33.1 Detailed Description 142
6.33.2 Typedef Documentation 142
6.33.2.1 clientID 142
6.33.3 Function Documentation 142
6.33.3.1 clientList_close 142
6.33.3.2 clientList_GetClientId 142
6.33.3.3 clientList_initialize 143
6.33.3.4 clientList_isClientAllowedToMakeAConnection 143
6.33.3.5 clientList_isClientAllowedToUseResource 143
6.33.3.6 clientList_isClientBanned 143
6.33.3.7 clientList_signalClientStoppedUsingResource 143
6.34 src/AmmServerlib/cache/dynamic_requests.c File Reference 144
6.34.1 Function Documentation 144
6.34.1.1 callClientRequestHandler 144
6.34.1.2 dynamicRequest_ContentAvailiable 144
6.34.1.3 dynamicRequest_serveContent 145
6.34.1.4 saveDynamicRequest 145
6.35 src/AmmServerlib/cache/dynamic_requests.h File Reference 145
6.35.1 Detailed Description 146
6.35.2 Function Documentation 146
6.35.2.1 callClientRequestHandler 146
6.35.2.2 dynamicRequest_ContentAvailiable 146

6.35.2.3 dynamicRequest_serveContent 147

6.35.2.4 saveDynamicRequest 147

6.36 src/AmmServerlib/cache/file_caching.c File Reference 148

6.36.1 Function Documentation 149

6.36.1.1 cache_AddDoNOTCacheRuleForResource 149

6.36.1.2 cache_AddFile 149

6.36.1.3 cache_AddMemoryBlock 150

6.36.1.4 cache_ChangeRequestIfTemplateRequested 150

6.36.1.5 cache_CountMemoryUsageAllocateOperation 150

6.36.1.6 cache_CountMemoryUsageFreeOperation 150

6.36.1.7 cache_CreateResource 151

6.36.1.8 cache_Destroy 151

6.36.1.9 cache_DestroyResource 151

6.36.1.10 cache_FindResource 151

6.36.1.11 cache_GetHashOfResource 151

6.36.1.12 cache_GetResource 152

6.36.1.13 cache_Initialize 152

6.36.1.14 cache_LoadResourceFromDisk 153

6.36.1.15 cache_RandomizeETAG 153

6.36.1.16 cache_RefreshResource 153

6.36.1.17 cache_RemoveContextAndResource 153

6.36.1.18 cache_RemoveResource 153

6.36.1.19 cache_ResourceExists 154

6.36.1.20 freeMallocIfNeeded 154

6.37 src/AmmServerlib/cache/file_caching.h File Reference 154

6.37.1 Detailed Description 155

6.37.2 Function Documentation 156

6.37.2.1 cache_AddDoNOTCacheRuleForResource 156

6.37.2.2 cache_AddFile 156

6.37.2.3 cache_AddMemoryBlock 156

6.37.2.4 cache_ChangeRequestIfTemplateRequested 157

6.37.2.5 cache_CountMemoryUsageAllocateOperation 157

6.37.2.6 cache_CountMemoryUsageFreeOperation 157

6.37.2.7 cache_Destroy 157

6.37.2.8 cache_FindResource 158

6.37.2.9 cache_GetHashOfResource 158

6.37.2.10 cache_GetResource 158

6.37.2.11 cache_Initialize 159

6.37.2.12 cache_RandomizeETAG 159

6.37.2.13 cache_RemoveContextAndResource 159

6.37.2.14 cache_RemoveResource 160
6.37.2.15 cache_ResourceExists 160
6.37.2.16 freeMallocIfNeeded 160
6.38 src/AmmServerlib/cache/file_compression.c File Reference 161
6.38.1 Function Documentation 161
6.38.1.1 CreateCompressedVersionofCachedResource 161
6.38.1.2 CreateCompressedVersionofDynamicContent 161
6.38.1.3 CreateCompressedVersionofStaticContent 162
6.38.1.4 CreateCompressedVersionofStaticContentPreloading 162
6.39 src/AmmServerlib/cache/file_compression.h File Reference 162
6.39.1 Detailed Description 163
6.39.2 Function Documentation 163
6.39.2.1 CreateCompressedVersionofCachedResource 163
6.39.2.2 CreateCompressedVersionofDynamicContent 163
6.39.2.3 CreateCompressedVersionofStaticContent 163
6.39.2.4 CreateCompressedVersionofStaticContentPreloading 164
6.40 src/AmmServerlib/hashmap/hashmap.c File Reference 164
6.40.1 Function Documentation 165
6.40.1.1 cmpHashTableItems 165
6.40.1.2 hashFunction 165
6.40.1.3 hashMap_Add 165
6.40.1.4 hashMap_AddULong 165
6.40.1.5 hashMap_Clear 166
6.40.1.6 hashMap_ContainsKey 166
6.40.1.7 hashMap_ContainsValue 166
6.40.1.8 hashMap_Create 166
6.40.1.9 hashMap_Destroy 167
6.40.1.10 hashMap_FindIndex 167
6.40.1.11 hashMap_GetCurrentNumberOfEntries 167
6.40.1.12 hashMap_GetHashAtIndex 167
6.40.1.13 hashMap_GetKeyAtIndex 168
6.40.1.14 hashMap_GetMaxNumberOfEntries 168
6.40.1.15 hashMap_GetPayload 168
6.40.1.16 hashMap_GetULongPayload 168
6.40.1.17 hashMap_Grow 169
6.40.1.18 hashMap_IsOK 169
6.40.1.19 hashMap_IsSorted 169
6.40.1.20 hashMap_LoadToFile 169
6.40.1.21 hashMap_SaveToFile 169
6.40.1.22 hashMap_Sort 169

6.40.1.23 hashmap_SwapRecords . . . 170
6.41 src/AmmServerlib/hashmap/hashmap.h File Reference . . . 170
6.41.1 Detailed Description . . . 171
6.41.2 Macro Definition Documentation . . . 171
6.41.2.1 HASHMAP_BE_THREAD_SAFE . . . 171
6.41.3 Function Documentation . . . 172
6.41.3.1 hashFunction . . . 172
6.41.3.2 hashMap_Add . . . 172
6.41.3.3 hashMap_AddULong . . . 172
6.41.3.4 hashMap_Clear . . . 172
6.41.3.5 hashMap_ContainsKey . . . 173
6.41.3.6 hashMap_ContainsValue . . . 174
6.41.3.7 hashMap_Create . . . 174
6.41.3.8 hashMap_Destroy . . . 174
6.41.3.9 hashMap_FindIndex . . . 174
6.41.3.10 hashMap_GetCurrentNumberOfEntries . . . 175
6.41.3.11 hashMap_GetHashAtIndex . . . 175
6.41.3.12 hashMap_GetKeyAtIndex . . . 175
6.41.3.13 hashMap_GetMaxNumberOfEntries . . . 175
6.41.3.14 hashMap_GetPayload . . . 176
6.41.3.15 hashMap_GetULongPayload . . . 176
6.41.3.16 hashMap_LoadToFile . . . 176
6.41.3.17 hashMap_SaveToFile . . . 176
6.41.3.18 hashMap_Sort . . . 177
6.41.3.19 hashmap_SwapRecords . . . 177
6.42 src/AmmServerlib/header_analysis/http_header_analysis.c File Reference . . . 177
6.42.1 Macro Definition Documentation . . . 178
6.42.1.1 CR . . . 178
6.42.1.2 LF . . . 178
6.42.2 Function Documentation . . . 178
6.42.2.1 AnalyzeHTTPHeader . . . 178
6.42.2.2 AnalyzeHTTPLineRequest . . . 179
6.42.2.3 AppendPOSTRequestToHTTPHeader . . . 179
6.42.2.4 FreeHTTPHeader . . . 179
6.42.2.5 HTTPHeaderComplete . . . 179
6.42.2.6 HTTPHeaderIsPOST . . . 179
6.42.2.7 ProcessAuthorizationHTTPLine . . . 180
6.42.2.8 ProcessFirstHTTPLine . . . 180
6.42.2.9 ProcessRangeHTTPLine . . . 180
6.42.2.10 ReceiveHTTPHeader . . . 180

6.43 src/AmmServerlib/header_analysis/http_header_analysis.h File Reference 180
6.43.1 Detailed Description 181
6.43.2 Function Documentation 181
6.43.2.1 AnalyzeHTTPHeader 181
6.43.2.2 AppendPOSTRequestToHTTPHeader 181
6.43.2.3 FreeHTTPHeader 182
6.43.2.4 HTTPHeaderComplete 182
6.43.2.5 HTTPHeaderIsPOST 182
6.43.2.6 ReceiveHTTPHeader 182
6.44 src/AmmServerlib/header_analysis/post_header_analysis.c File Reference 183
6.44.1 Function Documentation 183
6.44.1.1 AnalyzePOSTLineRequest 183
6.45 src/AmmServerlib/header_analysis/post_header_analysis.h File Reference 184
6.45.1 Detailed Description 184
6.45.2 Function Documentation 184
6.45.2.1 AnalyzePOSTLineRequest 184
6.46 src/AmmServerlib/InputParser/InputParser.cpp File Reference 184
6.46.1 Function Documentation 185
6.46.1.1 Version 185
6.46.2 Variable Documentation 185
6.46.2.1 ver 185
6.47 src/AmmServerlib/InputParser/InputParser.h File Reference 185
6.48 src/AmmServerlib/InputParser/InputParser_C.c File Reference 185
6.48.1 Macro Definition Documentation 186
6.48.1.1 WARN_ABOUT_INCORRECTLY_ALLOCATED_STACK_STRINGS 186
6.48.2 Function Documentation 186
6.48.2.1 CheckDelimeterNumOk 186
6.48.2.2 CheckIPCOk 186
6.48.2.3 CheckWordNumOk 186
6.48.2.4 InputParser_ClearNonCharacters 186
6.48.2.5 InputParser_Create 186
6.48.2.6 InputParser_DefaultDelimeters 186
6.48.2.7 InputParser_Destroy 187
6.48.2.8 InputParser_GetDelimeter 187
6.48.2.9 InputParser_GetLowercaseWord 187
6.48.2.10 InputParser_GetUpcaseWord 187
6.48.2.11 InputParser_GetWord 187
6.48.2.12 InputParser_GetWordChar 187
6.48.2.13 InputParser_GetWordFloat 187
6.48.2.14 InputParser_GetWordInt 187

6.48.2.15 InputParser_GetWordLength 187
6.48.2.16 InputParser_SelfCheck 187
6.48.2.17 InputParser_SeperateWords 187
6.48.2.18 InputParser_SeperateWordsCC 187
6.48.2.19 InputParser_SeperateWordsUC 188
6.48.2.20 InputParser_SetDelimeter 188
6.48.2.21 InputParser_TrimCharacters 188
6.48.2.22 InputParser_TrimCharactersEnd 188
6.48.2.23 InputParser_TrimCharactersStart 188
6.48.2.24 InputParser_WordCompare 188
6.48.2.25 InputParser_WordCompareAuto 188
6.48.2.26 InputParser_WordCompareNoCase 188
6.48.2.27 InputParser_WordCompareNoCaseAuto 188
6.48.2.28 InputParserC_Version 188
6.48.2.29 Str2Int_internal 188
6.48.3 Variable Documentation 188
6.48.3.1 _ipc_ver 188
6.48.3.2 warningsAboutIncorrectlyAllocatedStackIssued 188
6.49 src/AmmServerlib/InputParser/InputParser_C.h File Reference 188
6.49.1 Macro Definition Documentation 190
6.49.1.1 CONTAINERS_MAX 190
6.49.1.2 DELIM_MAX_MAX 190
6.49.1.3 MAX_COMPLICITY 190
6.49.1.4 MAX_MEMORY 190
6.49.1.5 MAX_STRING 190
6.49.1.6 USE_SCANF 190
6.49.2 Function Documentation 190
6.49.2.1 CheckWordNumOk 190
6.49.2.2 InputParser_ClearNonCharacters 190
6.49.2.3 InputParser_Create 190
6.49.2.4 InputParser_DefaultDelimeters 190
6.49.2.5 InputParser_Destroy 190
6.49.2.6 InputParser_GetDelimeter 190
6.49.2.7 InputParser_GetLowercaseWord 190
6.49.2.8 InputParser_GetUpcaseWord 190
6.49.2.9 InputParser_GetWord 190
6.49.2.10 InputParser_GetWordChar 191
6.49.2.11 InputParser_GetWordFloat 191
6.49.2.12 InputParser_GetWordInt 191
6.49.2.13 InputParser_GetWordLength 191

6.49.2.14 InputParser_SelfCheck 191
6.49.2.15 InputParser_SeperateWords 191
6.49.2.16 InputParser_SeperateWordsCC 191
6.49.2.17 InputParser_SeperateWordsUC 191
6.49.2.18 InputParser_SetDelimeter 191
6.49.2.19 InputParser_TrimCharacters 191
6.49.2.20 InputParser_TrimCharactersEnd 191
6.49.2.21 InputParser_TrimCharactersStart 191
6.49.2.22 InputParser_WordCompare 191
6.49.2.23 InputParser_WordCompareAuto 192
6.49.2.24 InputParser_WordCompareNoCase 192
6.49.2.25 InputParser_WordCompareNoCaseAuto 192
6.49.2.26 InputParserC_Version 192
6.50 src/AmmServerlib/network/file_server.c File Reference 192
6.50.1 Function Documentation 193
6.50.1.1 SendErrorFile 193
6.50.1.2 SendFile 193
6.50.1.3 SendMemoryBlockAsFile 193
6.50.1.4 SendPart 194
6.50.1.5 TransmitFileToSocket 194
6.50.1.6 TransmitFileToSocketInternal 194
6.50.2 Variable Documentation 194
6.50.2.1 files_open 194
6.51 src/AmmServerlib/network/file_server.h File Reference 194
6.51.1 Detailed Description 194
6.51.2 Function Documentation 195
6.51.2.1 SendErrorFile 195
6.51.2.2 SendFile 195
6.51.2.3 SendMemoryBlockAsFile 195
6.52 src/AmmServerlib/network/sendHTTPHeader.c File Reference 196
6.52.1 Function Documentation 196
6.52.1.1 SendAuthorizationHeader 196
6.52.1.2 SendErrorCodeHeader 197
6.52.1.3 SendNotModifiedHeader 197
6.52.1.4 SendSuccessCodeHeader 197
6.53 src/AmmServerlib/network/sendHTTPHeader.h File Reference 197
6.53.1 Detailed Description 198
6.53.2 Function Documentation 198
6.53.2.1 SendAuthorizationHeader 198
6.53.2.2 SendErrorCodeHeader 198

6.53.2.3 SendNotModifiedHeader 199
6.53.2.4 SendSuccessCodeHeader 200
6.54 src/AmmServerlib/server_configuration.c File Reference 200
6.54.1 Function Documentation 201
6.54.1.1 AssignStr 201
6.54.1.2 EmmitPossibleConfigurationWarnings 201
6.54.1.3 instance_CountFreeOP 202
6.54.1.4 instance_CountNewMallocOP 202
6.54.1.5 instance_WeCanCommitMoreMemory 202
6.54.1.6 LoadConfigurationFile 202
6.54.1.7 SetUsernameAndPassword 202
6.54.2 Variable Documentation 203
6.54.2.1 AccessLog 203
6.54.2.2 AccessLogEnable 203
6.54.2.3 CACHING_ENABLED 203
6.54.2.4 CHANGE_PRIORITY 203
6.54.2.5 CHANGE_TO_UID 203
6.54.2.6 ErrorLog 203
6.54.2.7 ErrorLogEnable 203
6.54.2.8 GLOBAL_KILL_SERVER_SWITCH 203
6.54.2.9 MAX_CACHE_SIZE_FOR_EACH_FILE_IN_MB 203
6.54.2.10 MAX_CACHE_SIZE_IN_MB 203
6.54.2.11 MAX_SEPERATE_CACHE_ITEMS 203
6.54.2.12 TemplatesInternalURI 203
6.54.2.13 USERNAME_UID_FOR_DAEMON 203
6.54.2.14 varSocketTimeoutREAD_seconds 204
6.54.2.15 varSocketTimeoutWRITE_seconds 204
6.55 src/AmmServerlib/server_configuration.h File Reference 204
6.55.1 Detailed Description 207
6.55.2 Macro Definition Documentation 207
6.55.2.1 CALCULATE_TIME_FOR_UPLOADS 207
6.55.2.2 CLIENT_SLEEP_TIME_INTERVAL_NSEC 207
6.55.2.3 CLIENT_SLEEP_TIME_WHEN_DYNAMIC_REQUEST_CALLBACK_IS_BUSY-_NSEC 207
6.55.2.4 COMPILE_WITH_CLIENT_LIST 207
6.55.2.5 DEFAULT_SOCKET_READ_TIMEOUT_SECS 207
6.55.2.6 DEFAULT_SOCKET_WRITE_TIMEOUT_SECS 207
6.55.2.7 DEFAULT_USERNAME_UID_FOR_DAEMON 208
6.55.2.8 DELAY_TRY_BINDING_TO_PORT 208
6.55.2.9 ENABLE_AUTOMATIC_CONFIGURATION_LOADING 208

6.55.2.10 ENABLE_COMPRESSION . . . 208
6.55.2.11 ENABLE_DIRECTORY_LISTING . . . 208
6.55.2.12 ENABLE_DROPPING_ROOT_UID_IF_ROOT . . . 208
6.55.2.13 ENABLE_DROPPING_UID_ALWAYS . . . 208
6.55.2.14 ENABLE_DYNAMIC_CONTENT_COMPRESSION . . . 208
6.55.2.15 ENABLE_INTERNAL_RESOURCES_RESOLVE . . . 208
6.55.2.16 ENABLE_POST . . . 208
6.55.2.17 EPOCH_YEAR_IN_TM_YEAR . . . 208
6.55.2.18 GROWSTEP_DIRECTORY_LIST_RESPONSE_BODY . . . 209
6.55.2.19 HTTP_POST_GROWTH_STEP_REQUEST_HEADER . . . 209
6.55.2.20 INITIAL_DIRECTORY_LIST_RESPONSE_BODY . . . 209
6.55.2.21 MAX_CLIENT_PRESPAWNED_THREADS . . . 209
6.55.2.22 MAX_CLIENT_THREADS . . . 209
6.55.2.23 MAX_CLIENTS_LISTENING_FOR . . . 209
6.55.2.24 MAX_CLIENTS_PER_IP . . . 209
6.55.2.25 MAX_CONFIGURATION_FILE_LINE_SIZE . . . 209
6.55.2.26 MAX_CONTENT_TYPE . . . 209
6.55.2.27 MAX_DIRECTORY_LIST_RESPONSE_BODY . . . 209
6.55.2.28 MAX_ETAG_SIZE . . . 209
6.55.2.29 MAX_FILE_READ_BLOCK_KB . . . 210
6.55.2.30 MAX_HTTP_POST_REQUEST_HEADER . . . 210
6.55.2.31 MAX_HTTP_REQUEST_HEADER . . . 210
6.55.2.32 MAX_HTTP_REQUEST_HEADER_LINES . . . 210
6.55.2.33 MAX_HTTP_REQUEST_HEADER_REPLY . . . 210
6.55.2.34 MAX_HTTP_REQUEST_SHORT_HEADER_REPLY . . . 210
6.55.2.35 MAX_RESOURCE_SLASHES . . . 210
6.55.2.36 MAX_TRIES_TO_BIND_TO_PORT . . . 210
6.55.2.37 NON_ROOT_UID_IF_USER_FAILS . . . 210
6.55.2.38 RANDOMIZE_ETAG_PER_LAUNCH . . . 210
6.55.2.39 REALLOC_TO_SAVE_MORE_THAN_THIS_NUMBER_BYTES . . . 210
6.55.2.40 TEMPLATE_INTERNAL_URI . . . 211
6.55.2.41 THREAD_MAXIMUM_TIME_TO_WAIT_FOR_A_NEWLY_CREATED_THREAD_MS . . . 211
6.55.2.42 THREAD_SLEEP_TIME_FOR_PRESPAWNED_THREADS . . . 211
6.55.2.43 THREAD_SLEEP_TIME_WHEN_OUR_PRESPAWNED_THREAD_IS_NEXT . . . 211
6.55.2.44 THREAD_SLEEP_TIME_WHILE_WAITING_FOR_NEW_CREATED_THREAD_TO_CONSUME_PARAMETERS . . . 211
6.55.2.45 WORKAROUND_REALLOCATION_R_X86_64_PC32_GCC_ERROR . . . 211
6.55.3 Function Documentation . . . 211
6.55.3.1 AssignStr . . . 211

6.55.3.2 EmmitPossibleConfigurationWarnings 211
6.55.3.3 instance_CountFreeOP 211
6.55.3.4 instance_CountNewMallocOP 212
6.55.3.5 instance_WeCanCommitMoreMemory 212
6.55.3.6 LoadConfigurationFile 212
6.55.3.7 SetUsernameAndPassword 212
6.55.4 Variable Documentation 213
6.55.4.1 AccessLog 213
6.55.4.2 AccessLogEnable 213
6.55.4.3 CACHING_ENABLED 213
6.55.4.4 CHANGE_PRIORITY 213
6.55.4.5 CHANGE_TO_UID 213
6.55.4.6 ErrorLog 213
6.55.4.7 ErrorLogEnable 213
6.55.4.8 GLOBAL_KILL_SERVER_SWITCH 213
6.55.4.9 MAX_CACHE_SIZE_FOR_EACH_FILE_IN_MB 213
6.55.4.10 MAX_CACHE_SIZE_IN_MB 213
6.55.4.11 MAX_SEPERATE_CACHE_ITEMS 213
6.55.4.12 TemplatesInternalURI 213
6.55.4.13 USERNAME_UID_FOR_DAEMON 213
6.55.4.14 varSocketTimeoutREAD_seconds 214
6.55.4.15 varSocketTimeoutWRITE_seconds 214
6.56 src/AmmServerlib/stringscanners/applicationFiles.c File Reference 214
6.56.1 Function Documentation 214
6.56.1.1 scanFor_applicationFiles 214
6.57 src/AmmServerlib/stringscanners/applicationFiles.h File Reference 214
6.57.1 Detailed Description 215
6.57.2 Enumeration Type Documentation 215
6.57.2.1 anonymous enum 215
6.57.3 Function Documentation 215
6.57.3.1 scanFor_applicationFiles 215
6.58 src/AmmServerlib/stringscanners/archiveFiles.c File Reference 215
6.58.1 Function Documentation 216
6.58.1.1 scanFor_archiveFiles 216
6.59 src/AmmServerlib/stringscanners/archiveFiles.h File Reference 216
6.59.1 Detailed Description 216
6.59.2 Enumeration Type Documentation 216
6.59.2.1 anonymous enum 216
6.59.3 Function Documentation 217
6.59.3.1 scanFor_archiveFiles 217

6.60 src/AmmServerlib/stringscanners/audioFiles.c File Reference 217
6.60.1 Function Documentation 218
6.60.1.1 scanFor_audioFiles 218
6.61 src/AmmServerlib/stringscanners/audioFiles.h File Reference 218
6.61.1 Detailed Description 218
6.61.2 Enumeration Type Documentation 218
6.61.2.1 anonymous enum 218
6.61.3 Function Documentation 219
6.61.3.1 scanFor_audioFiles 219
6.62 src/AmmServerlib/stringscanners/firstLines.c File Reference 219
6.62.1 Function Documentation 219
6.62.1.1 scanFor_firstLines 219
6.63 src/AmmServerlib/stringscanners/firstLines.h File Reference 220
6.63.1 Detailed Description 220
6.63.2 Enumeration Type Documentation 220
6.63.2.1 anonymous enum 220
6.63.3 Function Documentation 221
6.63.3.1 scanFor_firstLines 221
6.64 src/AmmServerlib/stringscanners/httpHeader.c File Reference 222
6.64.1 Function Documentation 222
6.64.1.1 scanFor_httpHeader 222
6.65 src/AmmServerlib/stringscanners/httpHeader.h File Reference 222
6.65.1 Detailed Description 223
6.65.2 Enumeration Type Documentation 223
6.65.2.1 anonymous enum 223
6.65.3 Function Documentation 223
6.65.3.1 scanFor_httpHeader 223
6.66 src/AmmServerlib/stringscanners/imageFiles.c File Reference 224
6.66.1 Function Documentation 224
6.66.1.1 scanFor_imageFiles 224
6.67 src/AmmServerlib/stringscanners/imageFiles.h File Reference 224
6.67.1 Detailed Description 225
6.67.2 Enumeration Type Documentation 225
6.67.2.1 anonymous enum 225
6.67.3 Function Documentation 225
6.67.3.1 scanFor_imageFiles 225
6.68 src/AmmServerlib/stringscanners/postHeader.c File Reference 226
6.68.1 Function Documentation 226
6.68.1.1 scanFor_postHeader 226
6.69 src/AmmServerlib/stringscanners/postHeader.h File Reference 226

6.69.1 Detailed Description 226
6.69.2 Enumeration Type Documentation 227
6.69.2.1 anonymous enum 227
6.69.3 Function Documentation 227
6.69.3.1 scanFor_postHeader 227
6.70 src/AmmServerlib/stringscanners/textFiles.c File Reference 227
6.70.1 Function Documentation 227
6.70.1.1 scanFor_textFiles 227
6.71 src/AmmServerlib/stringscanners/textFiles.h File Reference 228
6.71.1 Detailed Description 228
6.71.2 Enumeration Type Documentation 228
6.71.2.1 anonymous enum 228
6.71.3 Function Documentation 229
6.71.3.1 scanFor_textFiles 229
6.72 src/AmmServerlib/stringscanners/videoFiles.c File Reference 229
6.72.1 Function Documentation 229
6.72.1.1 scanFor_videoFiles 229
6.73 src/AmmServerlib/stringscanners/videoFiles.h File Reference 229
6.73.1 Detailed Description 230
6.73.2 Enumeration Type Documentation 230
6.73.2.1 anonymous enum 230
6.73.3 Function Documentation 230
6.73.3.1 scanFor_videoFiles 230
6.74 src/AmmServerlib/threads/clientServer.c File Reference 231
6.74.1 Function Documentation 231
6.74.1.1 ServeClient 231
6.74.1.2 ServeClientKeepAliveLoop 232
6.75 src/AmmServerlib/threads/clientServer.h File Reference 232
6.75.1 Detailed Description 232
6.75.2 Function Documentation 232
6.75.2.1 ServeClient 232
6.76 src/AmmServerlib/threads/freshThreads.c File Reference 233
6.76.1 Macro Definition Documentation 233
6.76.1.1 MAX_TRIES_TO_FIND_A_THREAD_ID 233
6.76.1.2 WEIRD_THING_THAT_WORKS 233
6.76.2 Function Documentation 233
6.76.2.1 FindAProperThreadID 233
6.76.2.2 SpawnThreadToServeNewClient 233
6.77 src/AmmServerlib/threads/freshThreads.h File Reference 234
6.77.1 Detailed Description 234

6.77.2 Function Documentation . . . 234
6.77.2.1 SpawnThreadToServeNewClient . . . 234
6.78 src/AmmServerlib/threads/prespawnedThreads.c File Reference . . . 235
6.78.1 Function Documentation . . . 235
6.78.1.1 PreSpawnedThread . . . 235
6.78.1.2 PreSpawnThreads . . . 235
6.78.1.3 UsePreSpawnedThreadToServeNewClient . . . 236
6.79 src/AmmServerlib/threads/prespawnedThreads.h File Reference . . . 236
6.79.1 Detailed Description . . . 237
6.79.2 Function Documentation . . . 237
6.79.2.1 PreSpawnThreads . . . 237
6.79.2.2 UsePreSpawnedThreadToServeNewClient . . . 237
6.80 src/AmmServerlib/threads/threadedServer.c File Reference . . . 237
6.80.1 Function Documentation . . . 238
6.80.1.1 HTTPServerIsRunning . . . 238
6.80.1.2 MainHTTPServerThread . . . 238
6.80.1.3 StartHTTPServer . . . 239
6.80.1.4 StopHTTPServer . . . 240
6.81 src/AmmServerlib/threads/threadedServer.h File Reference . . . 240
6.81.1 Detailed Description . . . 240
6.81.2 Function Documentation . . . 241
6.81.2.1 HTTPServerIsRunning . . . 241
6.81.2.2 StartHTTPServer . . . 242
6.81.2.3 StopHTTPServer . . . 242
6.82 src/AmmServerlib/threads/threadInitHelper.c File Reference . . . 242
6.82.1 Macro Definition Documentation . . . 243
6.82.1.1 SLEEP_FOR_N_NANOSECONDS_WAITING_STACK_MESSAGE . . . 243
6.83 src/AmmServerlib/threads/threadInitHelper.h File Reference . . . 243
6.83.1 Detailed Description . . . 243
6.84 src/AmmServerlib/tools/directory_lists.c File Reference . . . 243
6.84.1 Macro Definition Documentation . . . 244
6.84.1.1 tag_after_image . . . 244
6.84.1.2 tag_pre_image . . . 244
6.84.2 Function Documentation . . . 244
6.84.2.1 GenerateDirectoryPage . . . 244
6.84.2.2 path_cat . . . 244
6.84.3 Variable Documentation . . . 244
6.84.3.1 ending . . . 244
6.84.3.2 starting . . . 244
6.85 src/AmmServerlib/tools/directory_lists.h File Reference . . . 244

6.85.1 Detailed Description 244
6.85.2 Function Documentation 245
6.85.2.1 GenerateDirectoryPage 245
6.86 src/AmmServerlib/tools/http_tools.c File Reference 246
6.86.1 Function Documentation 247
6.86.1.1 CheckHTTPHeaderCategory 247
6.86.1.2 CheckHTTPHeaderCategoryAllCaps 247
6.86.1.3 CheckIfFileIsVideo 248
6.86.1.4 convertToUpperCase 248
6.86.1.5 DirectoryExistsAmmServ 248
6.86.1.6 encodeToBase64 248
6.86.1.7 FileExistsAmmServ 248
6.86.1.8 FilenameStripperOk 248
6.86.1.9 FindIndexFile 249
6.86.1.10 findOutClientIDOfPeer 249
6.86.1.11 freeString 249
6.86.1.12 GetContentType 249
6.86.1.13 GetContentTypeForExtension 249
6.86.1.14 GetExtensionImage 250
6.86.1.15 GetExtentionType 251
6.86.1.16 GetIntFromHTTPHeaderFieldPayload 251
6.86.1.17 GetNewStringFromHTTPHeaderFieldPayload 251
6.86.1.18 ReducePathSlashes_Inplace 251
6.86.1.19 RequestHTTPWebPage 251
6.86.1.20 seek_blank_char 252
6.86.1.21 seek_non_blank_char 252
6.86.1.22 ServerThreads_DropRootUID 252
6.86.1.23 setSocketTimeouts 252
6.86.1.24 StripGETRequestQueryAndFragment 252
6.86.1.25 StripHTMLCharacters_Inplace 252
6.86.1.26 StripVariableFromGETorPOSTString 252
6.86.1.27 stristr 253
6.86.1.28 stristr2Caps 253
6.86.1.29 strToUpcase 253
6.86.1.30 trim_last_empty_chars 253
6.87 src/AmmServerlib/tools/http_tools.h File Reference 253
6.87.1 Detailed Description 254
6.87.2 Typedef Documentation 254
6.87.2.1 contentType 254
6.87.3 Enumeration Type Documentation 254

6.87.3.1 contentTypeEnumerator 254
6.87.4 Function Documentation 255
6.87.4.1 CheckHTTPHeaderCategory 255
6.87.4.2 CheckHTTPHeaderCategoryAllCaps 255
6.87.4.3 CheckIfFileIsVideo 255
6.87.4.4 DirectoryExistsAmmServ 255
6.87.4.5 encodeToBase64 255
6.87.4.6 FileExistsAmmServ 255
6.87.4.7 FilenameStripperOk 256
6.87.4.8 FindIndexFile 256
6.87.4.9 findOutClientIDOfPeer 256
6.87.4.10 freeString 256
6.87.4.11 GetContentType 256
6.87.4.12 GetExtensionImage 257
6.87.4.13 GetExtentionType 257
6.87.4.14 GetIntFromHTTPHeaderFieldPayload 257
6.87.4.15 GetNewStringFromHTTPHeaderFieldPayload 257
6.87.4.16 ReducePathSlashes_Inplace 257
6.87.4.17 RequestHTTPWebPage 258
6.87.4.18 seek_blank_char 258
6.87.4.19 seek_non_blank_char 258
6.87.4.20 ServerThreads_DropRootUID 258
6.87.4.21 setSocketTimeouts 258
6.87.4.22 StripGETRequestQueryAndFragment 258
6.87.4.23 StripHTMLCharacters_Inplace 258
6.87.4.24 StripVariableFromGETorPOSTString 259
6.87.4.25 strToUpcase 259
6.87.4.26 trim_last_empty_chars 259
6.88 src/AmmServerlib/tools/logs.c File Reference 259
6.88.1 Function Documentation 259
6.88.1.1 AccessLogAppend 259
6.88.1.2 error 259
6.88.1.3 ErrorLogAppend 260
6.88.1.4 warning 260
6.89 src/AmmServerlib/tools/logs.h File Reference 260
6.89.1 Detailed Description 261
6.89.2 Macro Definition Documentation 261
6.89.2.1 BLACK 261
6.89.2.2 BLUE 261
6.89.2.3 BOLDBLACK 261

6.89.2.4 BOLDBLUE 261
6.89.2.5 BOLDCYAN 261
6.89.2.6 BOLDGREEN 261
6.89.2.7 BOLDMAGENTA 261
6.89.2.8 BOLDRED 261
6.89.2.9 BOLDWHITE 261
6.89.2.10 BOLDYELLOW 261
6.89.2.11 CYAN 261
6.89.2.12 GREEN 261
6.89.2.13 logEcho 261
6.89.2.14 MAGENTA 261
6.89.2.15 NORMAL 261
6.89.2.16 RED 261
6.89.2.17 WHITE 261
6.89.2.18 YELLOW 261
6.89.3 Function Documentation 261
6.89.3.1 AccessLogAppend 261
6.89.3.2 error 261
6.89.3.3 ErrorLogAppend 262
6.89.3.4 warning 262
6.90 src/AmmServerlib/tools/time_provider.c File Reference 262
6.90.1 Function Documentation 262
6.90.1.1 end_timer 262
6.90.1.2 GetDateString 263
6.90.1.3 GetTickCountAmmServ 263
6.90.1.4 start_timer 263
6.90.2 Variable Documentation 263
6.90.2.1 days 263
6.90.2.2 months 264
6.91 src/AmmServerlib/tools/time_provider.h File Reference 264
6.91.1 Detailed Description 264
6.91.2 Function Documentation 264
6.91.2.1 end_timer 264
6.91.2.2 GetDateString 264
6.91.2.3 GetTickCountAmmServ 265
6.91.2.4 start_timer 265
6.92 src/AmmServerlib/version.h File Reference 265
6.92.1 Macro Definition Documentation 265
6.92.1.1 RC_FILEVERSION 265
6.92.1.2 RC_FILEVERSION_STRING 265

6.93 src/Services/HabChan/board.c File Reference . . . 266
6.93.1 Function Documentation . . . 266
6.93.1.1 addBoardToSite . . . 266
6.93.1.2 loadBoardSettings . . . 266
6.93.1.3 prepareBoardIndexView . . . 266
6.94 src/Services/HabChan/board.h File Reference . . . 266
6.94.1 Function Documentation . . . 266
6.94.1.1 addBoardToSite . . . 266
6.94.1.2 prepareBoardIndexView . . . 267
6.95 src/Services/HabChan/main.h File Reference . . . 267
6.96 src/Services/HabChan/postReceiver.c File Reference . . . 267
6.96.1 Function Documentation . . . 267
6.96.1.1 processPostReceiver . . . 267
6.97 src/Services/HabChan/postReceiver.h File Reference . . . 267
6.97.1 Function Documentation . . . 267
6.97.1.1 processPostReceiver . . . 267
6.98 src/Services/HabChan/state.c File Reference . . . 268
6.98.1 Function Documentation . . . 268
6.98.1.1 addPostToThread . . . 268
6.98.1.2 debug_get_callback . . . 268
6.98.1.3 loadSite . . . 268
6.98.1.4 unloadSite . . . 268
6.98.2 Variable Documentation . . . 269
6.98.2.1 admin_server . . . 269
6.98.2.2 boardHashMap . . . 269
6.98.2.3 default_server . . . 269
6.98.2.4 GET_override . . . 269
6.98.2.5 ourSite . . . 269
6.98.2.6 threadHashMap . . . 269
6.98.2.7 threadIndexEndPage . . . 269
6.98.2.8 threadIndexEndPageLength . . . 269
6.98.2.9 threadIndexPage . . . 269
6.98.2.10 threadIndexPageLength . . . 269
6.98.2.11 threadIndexStartPage . . . 269
6.98.2.12 threadIndexStartPageLength . . . 269
6.99 src/Services/HabChan/state.h File Reference . . . 269
6.99.1 Macro Definition Documentation . . . 270
6.99.1.1 LINE_MAX_LENGTH . . . 270
6.99.1.2 MAX_BOARDS . . . 270
6.99.1.3 MAX_STRING_SIZE . . . 270

6.99.1.4 MAX_THREADS_PER_BOARD 270
6.99.2 Enumeration Type Documentation 270
6.99.2.1 FILETYPES_ENUM 270
6.99.3 Function Documentation 271
6.99.3.1 addPostToThread 271
6.99.3.2 loadSite 271
6.99.3.3 unloadSite 271
6.99.4 Variable Documentation 271
6.99.4.1 admin_server 271
6.99.4.2 boardHashMap 271
6.99.4.3 default_server 271
6.99.4.4 GET_override 271
6.99.4.5 ourSite 271
6.99.4.6 threadHashMap 271
6.99.4.7 threadIndexEndPage 271
6.99.4.8 threadIndexEndPageLength 271
6.99.4.9 threadIndexPage 271
6.99.4.10 threadIndexPageLength 271
6.99.4.11 threadIndexStartPage 271
6.99.4.12 threadIndexStartPageLength 271
6.100src/Services/HabChan/thread.c File Reference 271
6.100.1 Function Documentation 272
6.100.1.1 addThreadToBoard 272
6.100.1.2 loadThread 272
6.100.1.3 mallocHTMLListOfThreadsOfBoard 272
6.100.1.4 prepareThreadIndexView 272
6.100.1.5 prepareThreadView 272
6.101src/Services/HabChan/thread.h File Reference 272
6.101.1 Function Documentation 272
6.101.1.1 addThreadToBoard 272
6.101.1.2 prepareThreadIndexView 272
6.101.1.3 prepareThreadView 273
6.102src/Services/MyBlog/database.c File Reference 273
6.102.1 Function Documentation 273
6.102.1.1 appendPosts 273
6.102.1.2 loadPostsFromSQL 273
6.102.1.3 SQL_close 273
6.102.1.4 SQL_createInitialTables 273
6.102.1.5 SQL_error 273
6.102.1.6 SQL_getVersion 273

6.102.1.7 SQL_init 273
6.102.2 Variable Documentation 273
6.102.2.1 myblog 273
6.102.2.2 sqlserver 273
6.103src/Services/MyBlog/database.h File Reference 274
6.103.1 Macro Definition Documentation 274
6.103.1.1 CONTENT_BUFFER 274
6.103.1.2 MAX_CONTENT 274
6.103.1.3 MAX_MENU_ITEMS 274
6.103.1.4 MAX_STR 274
6.103.1.5 MAX_TAGS_PER_POST 275
6.103.1.6 MAX_WIDGET_ITEMS 275
6.103.2 Function Documentation 275
6.103.2.1 loadPostsFromSQL 275
6.103.2.2 SQL_close 275
6.103.2.3 SQL_createInitialTables 275
6.103.2.4 SQL_init 275
6.103.3 Variable Documentation 275
6.103.3.1 myblog 275
6.103.3.2 sqlserver 275
6.104src/Services/MyBlog/index.c File Reference 275
6.104.1 Function Documentation 276
6.104.1.1 destroy_index_prototype 276
6.104.1.2 getFooterLinksHTML 276
6.104.1.3 getLeftBlogRollHTML 276
6.104.1.4 getMenuListHTML 276
6.104.1.5 getPostListHTML 276
6.104.1.6 getRightBlogRollHTML 276
6.104.1.7 getWidgetListHTML 276
6.104.1.8 loadPosts 276
6.104.1.9 prepare_index 276
6.104.1.10prepare_index_prototype 276
6.104.1.11setupMyBlog 276
6.104.1.12strlimcpy 276
6.104.2 Variable Documentation 276
6.104.2.1 indexPage 276
6.105src/Services/MyBlog/index.h File Reference 276
6.105.1 Function Documentation 276
6.105.1.1 destroy_index_prototype 277
6.105.1.2 prepare_index 277

6.106src/Services/MyBlog/tools/myblogTool.c File Reference 277
6.106.1 Function Documentation 277
6.106.1.1 main 277
6.106.1.2 SQL_appendpost 277
6.106.1.3 SQL_close 277
6.106.1.4 SQL_error 277
6.106.1.5 SQL_getVersion 277
6.106.1.6 SQL_init 277
6.106.2 Variable Documentation 277
6.106.2.1 sqlserver 278
6.107src/Services/MyRemoteDesktop/xwd-1.0.5/clientwin.c File Reference 278
6.107.1 Function Documentation 278
6.107.1.1 Find_Client 278
6.108src/Services/MyRemoteDesktop/xwd-1.0.5/clientwin.h File Reference 278
6.108.1 Function Documentation 278
6.108.1.1 Find_Client 278
6.109src/Services/MyRemoteDesktop/xwd-1.0.5/config.h File Reference 278
6.109.1 Macro Definition Documentation 279
6.109.1.1 HAVE_INTTYPES_H 279
6.109.1.2 HAVE_MEMORY_H 279
6.109.1.3 HAVE_STDINT_H 279
6.109.1.4 HAVE_STDLIB_H 279
6.109.1.5 HAVE_STRING_H 279
6.109.1.6 HAVE_STRINGS_H 279
6.109.1.7 HAVE_SYS_STAT_H 279
6.109.1.8 HAVE_SYS_TYPES_H 279
6.109.1.9 HAVE_UNISTD_H 279
6.109.1.10PACKAGE 279
6.109.1.11PACKAGE_BUGREPORT 279
6.109.1.12PACKAGE_NAME 279
6.109.1.13PACKAGE_STRING 279
6.109.1.14PACKAGE_TARNAME 279
6.109.1.15PACKAGE_URL 279
6.109.1.16PACKAGE_VERSION 279
6.109.1.17PACKAGE_VERSION_MAJOR 279
6.109.1.18PACKAGE_VERSION_MINOR 279
6.109.1.19PACKAGE_VERSION_PATCHLEVEL 279
6.109.1.20STDC_HEADERS 279
6.109.1.21VERSION 279
6.110src/Services/MyRemoteDesktop/xwd-1.0.5/dsimple.c File Reference 280

6.110.1 Macro Definition Documentation 280
6.110.1.1 ARGC 280
6.110.1.2 COPYOPT 280
6.110.1.3 NXTOPT 280
6.110.1.4 NXTOPTP 280
6.110.1.5 OPTION 280
6.110.2 Function Documentation 281
6.110.2.1 Close_Display 281
6.110.2.2 Fatal_Error 281
6.110.2.3 Get_Display_Name 281
6.110.2.4 getRootWindow 281
6.110.2.5 Open_Display 281
6.110.2.6 outl 281
6.110.2.7 Select_Window 281
6.110.2.8 Select_Window_Args 281
6.110.2.9 Setup_Display_And_Screen 281
6.110.2.10Setup_Null_Display_And_Screen 281
6.110.2.11Window_With_Name 281
6.110.3 Variable Documentation 281
6.110.3.1 dpy 281
6.110.3.2 program_name 281
6.110.3.3 screen 281
6.111src/Services/MyRemoteDesktop/xwd-1.0.5/dsimple.h File Reference 281
6.111.1 Macro Definition Documentation 282
6.111.1.1 INIT_NAME 282
6.111.1.2 MAX 282
6.111.1.3 MIN 282
6.111.1.4 X_USAGE 282
6.111.2 Function Documentation 282
6.111.2.1 Close_Display 282
6.111.2.2 Fatal_Error 282
6.111.2.3 Get_Display_Name 283
6.111.2.4 getRootWindow 283
6.111.2.5 Open_Display 283
6.111.2.6 outl 283
6.111.2.7 Select_Window 283
6.111.2.8 Select_Window_Args 283
6.111.2.9 Setup_Display_And_Screen 283
6.111.2.10Setup_Null_Display_And_Screen 283
6.111.2.11usage 283

6.111.2.12Window_With_Name 283
6.111.3 Variable Documentation 283
6.111.3.1 dpy 283
6.111.3.2 program_name 283
6.111.3.3 screen 283
6.112src/Services/MyRemoteDesktop/xwd-1.0.5/list.c File Reference 283
6.112.1 Function Documentation 284
6.112.1.1 add_to_list 284
6.112.1.2 delete_from_list 284
6.112.1.3 delete_list 284
6.112.1.4 delete_list_destroying 284
6.112.1.5 dup_list_head 284
6.112.1.6 first_in_list 284
6.112.1.7 list_is_empty 285
6.112.1.8 list_length 285
6.112.1.9 new_list 285
6.112.1.10next_in_list 285
6.112.1.11zero_list 285
6.113src/Services/MyRemoteDesktop/xwd-1.0.5/list.h File Reference 285
6.113.1 Macro Definition Documentation 286
6.113.1.1 DUP_WHOLE_LIST 286
6.113.1.2 EQUAL 286
6.113.1.3 GREATER 286
6.113.1.4 LESS 286
6.113.1.5 START_AT_CURR 287
6.113.2 Typedef Documentation 287
6.113.2.1 DESTRUCT_FUNC_PTR 287
6.113.2.2 list 287
6.113.2.3 list_item 287
6.113.2.4 list_ptr 287
6.113.3 Function Documentation 287
6.113.3.1 add_to_list 287
6.113.3.2 delete_from_list 287
6.113.3.3 delete_list 287
6.113.3.4 delete_list_destroying 287
6.113.3.5 dup_list_head 288
6.113.3.6 first_in_list 288
6.113.3.7 list_is_empty 288
6.113.3.8 list_length 288
6.113.3.9 new_list 288

6.113.3.10next_in_list 288
6.113.3.11zero_list 288
6.114src/Services/MyRemoteDesktop/xwd-1.0.5/multiVis.c File Reference 289
6.114.1 Macro Definition Documentation 290
6.114.1.1 BLUE_SHIFT 290
6.114.1.2 DIRECT_COLOR 290
6.114.1.3 GRAY_SCALE 290
6.114.1.4 GREEN_SHIFT 290
6.114.1.5 MAX 290
6.114.1.6 MIN 290
6.114.1.7 PSEUDO_COLOR 290
6.114.1.8 RED_SHIFT 290
6.114.1.9 SAME_REGIONS 290
6.114.1.10STATIC_GRAY 290
6.114.1.11TRUE_COLOR 290
6.114.2 Typedef Documentation 290
6.114.2.1 myBOX 290
6.114.2.2 myBoxPtr 291
6.114.2.3 myBoxRec 291
6.114.2.4 myREGION 291
6.114.3 Function Documentation 291
6.114.3.1 FreeXVisualInfo 291
6.114.3.2 GetMultiVisualRegions 291
6.114.3.3 GetXVisualInfo 291
6.114.3.4 initFakeVisual 291
6.114.3.5 ReadAreaToImage 291
6.115src/Services/MyRemoteDesktop/xwd-1.0.5/multiVis.h File Reference 291
6.115.1 Function Documentation 292
6.115.1.1 GetMultiVisualRegions 292
6.115.1.2 initFakeVisual 292
6.115.1.3 ReadAreaToImage 292
6.116src/Services/MyRemoteDesktop/xwd-1.0.5/wsutils.h File Reference 292
6.116.1 Macro Definition Documentation 293
6.116.1.1 FLEXIBLE 293
6.116.1.2 None 293
6.116.1.3 NOT_FLEXIBLE 293
6.116.1.4 SB_CMAP_TYPE_FULL 293
6.116.1.5 SB_CMAP_TYPE_MONOTONIC 293
6.116.1.6 SB_CMAP_TYPE_NORMAL 293
6.116.2 Function Documentation 293

6.116.2.1 CreateImagePlanesWindow . . . 293
6.116.2.2 CreateOverlayPlanesWindow . . . 294
6.116.2.3 FindImagePlanesVisual . . . 294
6.116.2.4 FindOverlayPlanesVisual . . . 294
6.116.2.5 FreeXVisualInfo . . . 294
6.116.2.6 GetXVisualInfo . . . 294
6.117src/Services/MyRemoteDesktop/xwd-1.0.5/xwd.c File Reference . . . 294
6.117.1 Macro Definition Documentation . . . 295
6.117.1.1 FEEP_VOLUME . . . 295
6.117.1.2 lowbit . . . 295
6.117.2 Typedef Documentation . . . 295
6.117.2.1 Pixel . . . 295
6.117.3 Function Documentation . . . 295
6.117.3.1 _swaplong . . . 295
6.117.3.2 _swapshort . . . 295
6.117.3.3 Get_XColors . . . 295
6.117.3.4 Image_Size . . . 295
6.117.3.5 main . . . 295
6.117.3.6 usage . . . 295
6.117.3.7 Window_Dump . . . 295
6.118src/Services/MyRemoteDesktop/xwd-1.0.5/XwdLib.h File Reference . . . 295
6.118.1 Function Documentation . . . 295
6.118.1.1 closeXwdLib . . . 295
6.118.1.2 getScreen . . . 295
6.118.1.3 initXwdLib . . . 295
6.119src/Services/MyTube/indexer.c File Reference . . . 296
6.119.1 Macro Definition Documentation . . . 296
6.119.1.1 DEFAULT_TEST_TRANSMISSION_VIDEO_TITLE . . . 296
6.119.2 Function Documentation . . . 296
6.119.2.1 clearExtensionFAST . . . 296
6.119.2.2 getAVideoForQuery . . . 296
6.119.2.3 loadVideoDatabase . . . 296
6.119.2.4 path_cat2 . . . 296
6.119.3 Variable Documentation . . . 296
6.119.3.1 videoDefaultTestTranmission . . . 296
6.120src/Services/MyTube/indexer.h File Reference . . . 296
6.120.1 Macro Definition Documentation . . . 297
6.120.1.1 MAX_STR . . . 297
6.120.2 Function Documentation . . . 297
6.120.2.1 loadVideoDatabase . . . 297

6.120.2.2 path_cat2 297

6.120.3 Variable Documentation 297

6.120.3.1 videoDefaultTestTranmission 297

6.121 src/Services/MyTube/thumbnailer.c File Reference 297

6.121.1 Function Documentation 298

6.121.1.1 generateThumbnailOfVideo 298

6.122 src/Services/MyTube/thumbnailer.h File Reference 298

6.122.1 Macro Definition Documentation 298

6.122.1.1 GENERATE_NEW_THUMBNAILS_LIVE 298

6.122.2 Function Documentation 298

6.122.2.1 generateThumbnailOfVideo 298

6.123 src/Services/ScriptRunner/main.cpp File Reference 298

6.123.1 Macro Definition Documentation 299

6.123.1.1 ADMIN_BINDING_PORT 299

6.123.1.2 DEFAULT_BINDING_PORT 299

6.123.1.3 ENABLE_ADMIN_PAGE 299

6.123.1.4 ENABLE_CHAT_BOX 299

6.123.1.5 ENABLE_PASSWORD_PROTECTION 299

6.123.1.6 MAX_BINDING_PORT 299

6.123.1.7 MAX_COMMAND_SIZE 300

6.123.2 Function Documentation 300

6.123.2.1 close_dynamic_content 300

6.123.2.2 EraseFile 300

6.123.2.3 execute 300

6.123.2.4 FileExistsTest 300

6.123.2.5 getBackCommandLine 300

6.123.2.6 init_dynamic_content 300

6.123.2.7 joystickExecute 300

6.123.2.8 main 300

6.123.2.9 prepare_base_image 300

6.123.2.10prepare_form_content_callback 300

6.123.2.11prepare_index_content_callback 300

6.123.2.12prepare_stats_content_callback 300

6.123.2.13prepare_top_image 300

6.123.2.14replaceChar 301

6.123.2.15store_new_configuration_callback 301

6.123.2.16StringIsHTMLSafe 301

6.123.2.17termination_handler 301

6.123.3 Variable Documentation 301

6.123.3.1 admin_root 301

6.123.3.2 admin_server 301
6.123.3.3 base_image 301
6.123.3.4 chatbox 301
6.123.3.5 default_server 301
6.123.3.6 form 301
6.123.3.7 GET_override 301
6.123.3.8 indexPage 301
6.123.3.9 page 301
6.123.3.10pageLength 301
6.123.3.11random_chars 301
6.123.3.12settings 301
6.123.3.13stats 301
6.123.3.14templates_root 301
6.123.3.15top_image 301
6.123.3.16webserver_root 301
6.124src/Services/SQLiteServer/sqlite.c File Reference 302
6.124.1 Function Documentation 302
6.124.1.1 printCars 302
6.124.1.2 serveCarsPageWithSQL 302
6.124.1.3 SQL_close 302
6.124.1.4 SQL_getVersion 302
6.124.1.5 SQL_init 302
6.124.1.6 SQL_populate 302
6.125src/Services/SQLiteServer/sqlite.h File Reference 302
6.125.1 Function Documentation 303
6.125.1.1 serveCarsPageWithSQL 303
6.125.1.2 SQL_close 303
6.125.1.3 SQL_getVersion 303
6.125.1.4 SQL_init 303
6.125.1.5 SQL_populate 303
6.126src/StringRecognizer/fastStringParser.c File Reference 303
6.126.1 Macro Definition Documentation 304
6.126.1.1 ACTIVATED_LEVELS 304
6.126.1.2 MAXIMUM_FILENAME_WITH_EXTENSION 304
6.126.1.3 MAXIMUM_LEVELS 304
6.126.1.4 MAXIMUM_LINE_LENGTH 304
6.126.2 Function Documentation 304
6.126.2.1 addLevelSpaces 304
6.126.2.2 convertTo_ENUM_ID 304
6.126.2.3 export_C_Scanner 304

6.126.2.4 fastStringParser_addString 304
6.126.2.5 fastStringParser_close 304
6.126.2.6 fastStringParser_countStringsForNextChar 305
6.126.2.7 fastSTringParser_createRulesFromFile 305
6.126.2.8 fastStringParser_hasStringsWithNConsecutiveChars 305
6.126.2.9 fastStringParser_initialize 305
6.126.2.10printAllEnumeratorItems 305
6.126.2.11printIfAllPossibleStrings 305
6.126.2.12recursiveTraverser 305
6.126.3 Variable Documentation 305
6.126.3.1 acceptedChars 305
6.126.3.2 fspHTTPHeader 305
6.127src/StringRecognizer/fastStringParser.h File Reference 305
6.127.1 Detailed Description 306
6.127.2 Function Documentation 306
6.127.2.1 export_C_Scanner 306
6.127.2.2 fastStringParser_close 306
6.127.2.3 fastSTringParser_createRulesFromFile 307
6.128src/UnitTests/testHashMap.c File Reference 307
6.128.1 Macro Definition Documentation 307
6.128.1.1 BLACK 307
6.128.1.2 BLUE 307
6.128.1.3 CYAN 307
6.128.1.4 GREEN 307
6.128.1.5 MAGENTA 308
6.128.1.6 NORMAL 308
6.128.1.7 RED 308
6.128.1.8 WHITE 308
6.128.1.9 YELLOW 308
6.128.2 Function Documentation 308
6.128.2.1 doHashMapTest 308
6.128.2.2 doInjectTest 308
6.128.2.3 main 308
6.129src/UserAccounts/userAccounts.h File Reference 308
6.129.1 Typedef Documentation 309
6.129.1.1 UserAccount_PasswordEncoding 309
6.129.1.2 UserAccount_UserID 309
6.129.2 Enumeration Type Documentation 309
6.129.2.1 UserAccountPasswordEncodingEnum 309
6.129.3 Function Documentation 309

6.129.3.1 uadb_authenticateUser 309
6.129.3.2 uadb_closeUserAccountDatabase 309
6.129.3.3 uadb_initializeUserAccountDatabase 309
6.129.3.4 uadb_loginUser 309

Index **310**

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/AmmarServer

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..

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)

- 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

The projects build dependencies are the gcc compiler , pthreads and pretty basic things (see https://raw.githubusercontent.com/AmmarkoV/AmmarServer/master/doc/dependencies.jpg) which is one of the core reasons of the "beauty" of this project :)

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 ENABLE_COMPRESSION at server_configuration.h 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

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 (const char ∗command, char ∗what2GetBack, unsigned int what2GetBackMaxSize)

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

Global AmmServer_ExecuteCommandLineNum (const char ∗command, char ∗what2GetBack, unsigned int what2GetBackMaxSize, unsigned int lineNumber)

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

Global AmmServer_ReplaceAllVarsInMemoryFile (char ∗page, unsigned int instances, unsigned int pageLength, const char ∗var, const char ∗value)

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

Global AmmServer_ReplaceVarInMemoryFile (char ∗page, unsigned int pageLength, const char ∗var, const char ∗value)

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

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 astringInjectDataToMemoryHandler (struct AmmServer_MemoryHandler ∗mh, const char ∗var, const char ∗value)

This does not yet reallocate the buffer to make it bigger in case it is not big enough to accomodate the new string..

Global cache_CountMemoryUsageAllocateOperation (struct AmmServer_Instance ∗instance, unsigned long allocatedSize)

cache_CountMemoryUsageAllocateOperation should have a mutex lock so that it is well defined on massively parallel operations

Return values

1=Ok,0=Failed	

Global cache_CountMemoryUsageFreeOperation (struct AmmServer_Instance ∗instance, unsigned long freedSize)

cache_CountMemoryUsageFreeOperation should have a mutex lock so that it is well defined on massively parallel operations

Return values

1=Ok,0=Failed	

Global cache_GetResource (struct AmmServer_Instance ∗instance, struct HTTPHeader ∗request, unsigned int resourceCacheID, char ∗verified_filename, unsigned int verified_filenameSize, unsigned int ∗index, unsigned long ∗filesize, struct stat ∗last_modification, unsigned char ∗compressionSupported, unsigned char ∗freeContentAfterUsingIt, unsigned char ∗serveAsRegularFile)

This function should check filesizes/dates and refresh memory snapshots 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

Global dynamicRequest_serveContent (struct AmmServer_Instance ∗instance, struct HTTPHeader ∗request, struct AmmServer_RH_Context ∗shared_context, char ∗verified_filename, unsigned int verified_filenameLength, unsigned int index, unsigned long ∗memSize, unsigned char ∗compressionSupported, unsigned char ∗freeContentAfterUsingIt, unsigned char ∗contentContainsPathToFileToBeStreamed)

Current implementation waits for new content , should add content double buffering to always have a valid buffer and serve it instantly , `https://github.com/AmmarkoV/AmmarServer/issues/28`

Global EmmitPossibleConfigurationWarnings (struct AmmServer_Instance ∗instance)

TOP PRIORITY -> Implement POST !FILE! requests , and couple them to dynamic content

Implement download resume capabilities (range head request) ..

require the Host: header from HTTP 1.1 clients

accept absolute URL's in a request

accept requests with chunked data

use the "100 Continue" response appropriately

handle requests with If-Modified-Since: or If-Unmodified-Since: headers

Add configuration file ammServ.conf parsing..

Add detailed input header parsing

Improve directory listings (add filesizes , dates etc)

Improve implemented file caching mechanism (add string comparison to make code hash collision free)

Add apache like logging capabilities

File fastStringParser.h

In case the declarations have shared prefixes and the shortest prefix is stated first they will also get recognized first so be careful

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 , needs a lot of work

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, const 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 ?

Global ProcessRangeHTTPLine (char ∗request, unsigned int requestLength, unsigned long ∗rangeStart, unsigned long ∗rangeEnd)

: ProcessRangeHTTPLine , can be improved , it is not thoroughly tested

Global ReceiveHTTPHeader (struct AmmServer_Instance ∗instance, int clientSock, unsigned long ∗headerLength)

Reallocation code of ReceiveHTTPHeader when we jump from a regular GET memory block to a large POST memory block is shit and needs to be fixed

Global RequestHTTPWebPage (char ∗hostname, unsigned int port, char ∗filename, unsigned int max_content)

: Check for success or failure on RequestHTTPWebPage and return an appropriate return value

Global SendErrorCodeHeader (int clientsock, unsigned int error_code, const char ∗verified_filename, const char ∗templates_root)

This call seems to fail ?

Return values

1=Success,0=Failure	

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 SpawnThreadToServeNewClient (struct AmmServer_Instance ∗instance, int clientsock, struct sockaddr_in client, unsigned int clientlen)

There might be issues with the way the compiler optimizes the code that waits for the stack to be read before continuing on from the main thread..

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 StripVariableFromGETorPOSTString (const char ∗input, const char ∗var_id, char ∗var_val, unsigned int var_val_length)

StripVariableFromGETorPOSTString does not have a high quality implementation

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:

_list_item 15

AmmServer_DynamicRequest
When a call to a function that is a dynamic request is done this is the structure that holds the information 15

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 16

AmmServer_Instance_Settings
Each Instance of AmmarServer has some basic settings , which are stored in AmmServer_Instance_Settings 18

AmmServer_MemoryHandler
A Wrapper around a memory buffer that enables house keeping for reallocations etc 19

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 19

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 20

board 21

cache_item
A cache item and all it's contents 22

clientListContext
The client list is just a hashmap (see hashmap.h) 23

fastStringParser
Internal Structure that holds all the string parser context 23

fspString
Internal Structure to hold a string and its id for further processing 24

guard_byte 25

hashMap
The central structure for the hash map 25

hashMapEntry
An entry on the hash map flattened out for ease of use 26

htmlContent 26

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 27

HTTPTransaction
Structure to keep data for an HTTP Transaction . . . 29
Image . . . 30
image_region_type . . . 30
image_win_type . . . 31
InputParser . . . 32
InputParserC . . . 34
linkItemList . . . 35
linkLabelItem . . . 35
menuItemList . . . 36
my_XRegion . . . 36
myBox . . . 37
OverlayInfo . . . 38
OverlayVisualPropertyRec . . . 38
PassToHTTPThread
A structure that holds information to be passed from the main thread to the new (fresh) thread . 39
PassToPreSpawnedThread . . . 40
playlist . . . 40
playlistItem . . . 41
post . . . 41
postItem . . . 42
postItemList . . . 43
PreSpawnedThread
A structure that holds information to be passed from the main thread to the new (prespawned) thread . . . 43
site . . . 44
socialLinks . . . 45
SQLiteSession . . . 45
tagItem . . . 46
tagItemList . . . 46
thread . . . 47
time_snap . . . 48
timestamp
Timestamp for a cache item entry . . . 48
tokens . . . 49
URLDB . . . 50
UserAccountAuthenticationToken . . . 50
UserAccountDatabase . . . 50
videoCollection . . . 51
videoItem . . . 51
website . . . 52
widgetItem . . . 53
widgetItemList . . . 53

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

doc/DoxygenMainpage.h 55
doc/helloworld.c 55
src/AmmCaptcha/AmmCaptcha.h 56
src/AmmCaptcha/imaging.c 110
src/AmmCaptcha/imaging.h 111
src/AmmCaptcha/img_warp.c 112
src/AmmCaptcha/img_warp.h 113
src/AmmCaptcha/jpgInput.c 113
src/AmmCaptcha/jpgInput.h 115
src/AmmCaptcha/main.c 57
src/AmmCaptcha/AmmCaptchaTester/main.c 57
src/AmmServerlib/AmmServerlib.h
The Main Header for AmmarServer 115
src/AmmServerlib/main.c 60
src/AmmServerlib/server_configuration.c 200
src/AmmServerlib/server_configuration.h
The Main Header for the settings used by AmmarServer 204
src/AmmServerlib/version.h 265
src/AmmServerlib/AString/AString.c 136
src/AmmServerlib/AString/AString.h
A small toolset to handle long strings manage memory and append,inject other strings inside them 138
src/AmmServerlib/cache/client_list.c 139
src/AmmServerlib/cache/client_list.h
Client list for IPs that should also serve as a banlist manage QoS etc 141
src/AmmServerlib/cache/dynamic_requests.c 144
src/AmmServerlib/cache/dynamic_requests.h
Dynamic request handler , one of the most important parts of this library 145
src/AmmServerlib/cache/file_caching.c 148
src/AmmServerlib/cache/file_caching.h
Central cache of AmmarServer , it reads/indexes and swaps resources asked by clients for fast performance 154
src/AmmServerlib/cache/file_compression.c 161
src/AmmServerlib/cache/file_compression.h
A tool that compresses memory blocks for better bandwidth usage on the expense of computing power 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 . . . 170
src/AmmServerlib/header_analysis/http_header_analysis.c . . . 177
src/AmmServerlib/header_analysis/http_header_analysis.h
Tools to process HTTP requests . . . 180
src/AmmServerlib/header_analysis/post_header_analysis.c . . . 183
src/AmmServerlib/header_analysis/post_header_analysis.h
Tools to process POST requests . . . 184
src/AmmServerlib/InputParser/InputParser.cpp . . . 184
src/AmmServerlib/InputParser/InputParser.h . . . 185
src/AmmServerlib/InputParser/InputParser_C.c . . . 185
src/AmmServerlib/InputParser/InputParser_C.h . . . 188
src/AmmServerlib/InputParser/InputParser_C_Tester/main.c . . . 59
src/AmmServerlib/network/file_server.c . . . 192
src/AmmServerlib/network/file_server.h
Basic file server functionality of AmmarServer . . . 194
src/AmmServerlib/network/sendHTTPHeader.c . . . 196
src/AmmServerlib/network/sendHTTPHeader.h
Small code segments that transmit HTTP responses . . . 197
src/AmmServerlib/stringscanners/applicationFiles.c . . . 214
src/AmmServerlib/stringscanners/applicationFiles.h
A tool that scans for a string in a very fast and robust way . . . 214
src/AmmServerlib/stringscanners/archiveFiles.c . . . 215
src/AmmServerlib/stringscanners/archiveFiles.h
A tool that scans for a string in a very fast and robust way . . . 216
src/AmmServerlib/stringscanners/audioFiles.c . . . 217
src/AmmServerlib/stringscanners/audioFiles.h
A tool that scans for a string in a very fast and robust way . . . 218
src/AmmServerlib/stringscanners/firstLines.c . . . 219
src/AmmServerlib/stringscanners/firstLines.h
A tool that scans for a string in a very fast and robust way . . . 220
src/AmmServerlib/stringscanners/httpHeader.c . . . 222
src/AmmServerlib/stringscanners/httpHeader.h
A tool that scans for a string in a very fast and robust way . . . 222
src/AmmServerlib/stringscanners/imageFiles.c . . . 224
src/AmmServerlib/stringscanners/imageFiles.h
A tool that scans for a string in a very fast and robust way . . . 224
src/AmmServerlib/stringscanners/postHeader.c . . . 226
src/AmmServerlib/stringscanners/postHeader.h
A tool that scans for a string in a very fast and robust way . . . 226
src/AmmServerlib/stringscanners/textFiles.c . . . 227
src/AmmServerlib/stringscanners/textFiles.h
A tool that scans for a string in a very fast and robust way . . . 228
src/AmmServerlib/stringscanners/videoFiles.c . . . 229
src/AmmServerlib/stringscanners/videoFiles.h
A tool that scans for a string in a very fast and robust way . . . 229
src/AmmServerlib/threads/clientServer.c . . . 231
src/AmmServerlib/threads/clientServer.h
This is the entry point to serve a client that picks a prespawned thread or creates a fresh new one and then handles the requests. . . . 232
src/AmmServerlib/threads/freshThreads.c . . . 233
src/AmmServerlib/threads/freshThreads.h
Creating new threads to serve clients , we only have one call that generates a thread that serves a client connection . . . 234
src/AmmServerlib/threads/prespawnedThreads.c . . . 235
src/AmmServerlib/threads/prespawnedThreads.h
Using already created threads to serve clients , we have a pool of threads that can be used to serve connections . . . 236

src/AmmServerlib/threads/threadedServer.c . . . 237
src/AmmServerlib/threads/threadedServer.h
Creating new threads to serve clients , we only have one call that generates a thread that serves a client connection . . . 240
src/AmmServerlib/threads/threadInitHelper.c . . . 242
src/AmmServerlib/threads/threadInitHelper.h
Helper Functions to help with passing messages around . . . 243
src/AmmServerlib/tools/directory_lists.c . . . 243
src/AmmServerlib/tools/directory_lists.h
Basic file server functionality of AmmarServer . . . 244
src/AmmServerlib/tools/http_tools.c . . . 246
src/AmmServerlib/tools/http_tools.h
A collection of tools required by the server and gathered here since they do a very specific job 253
src/AmmServerlib/tools/logs.c . . . 259
src/AmmServerlib/tools/logs.h
Logging functions . . . 260
src/AmmServerlib/tools/time_provider.c . . . 262
src/AmmServerlib/tools/time_provider.h
Timer functions . . . 264
src/ScriptRunner/main.c . . . 78
src/Services/AmmarServer/main.c . . . 82
src/Services/CinemaPilot/main.c . . . 85
src/Services/GeoPosShare/main.c . . . 88
src/Services/HabChan/board.c . . . 266
src/Services/HabChan/board.h . . . 266
src/Services/HabChan/main.c . . . 91
src/Services/HabChan/main.h . . . 267
src/Services/HabChan/postReceiver.c . . . 267
src/Services/HabChan/postReceiver.h . . . 267
src/Services/HabChan/state.c . . . 268
src/Services/HabChan/state.h . . . 269
src/Services/HabChan/thread.c . . . 271
src/Services/HabChan/thread.h . . . 272
src/Services/MyBlog/database.c . . . 273
src/Services/MyBlog/database.h . . . 274
src/Services/MyBlog/index.c . . . 275
src/Services/MyBlog/index.h . . . 276
src/Services/MyBlog/main.c . . . 92
src/Services/MyBlog/tools/myblogTool.c . . . 277
src/Services/MyLoader/main.c . . . 94
src/Services/MyRemoteDesktop/main.c . . . 95
src/Services/MyRemoteDesktop/xwd-1.0.5/clientwin.c . . . 278
src/Services/MyRemoteDesktop/xwd-1.0.5/clientwin.h . . . 278
src/Services/MyRemoteDesktop/xwd-1.0.5/config.h . . . 278
src/Services/MyRemoteDesktop/xwd-1.0.5/dsimple.c . . . 280
src/Services/MyRemoteDesktop/xwd-1.0.5/dsimple.h . . . 281
src/Services/MyRemoteDesktop/xwd-1.0.5/list.c . . . 283
src/Services/MyRemoteDesktop/xwd-1.0.5/list.h . . . 285
src/Services/MyRemoteDesktop/xwd-1.0.5/main.c . . . 97
src/Services/MyRemoteDesktop/xwd-1.0.5/multiVis.c . . . 289
src/Services/MyRemoteDesktop/xwd-1.0.5/multiVis.h . . . 291
src/Services/MyRemoteDesktop/xwd-1.0.5/wsutils.h . . . 292
src/Services/MyRemoteDesktop/xwd-1.0.5/xwd.c . . . 294
src/Services/MyRemoteDesktop/xwd-1.0.5/XwdLib.h . . . 295
src/Services/MyTube/indexer.c . . . 296
src/Services/MyTube/indexer.h . . . 296
src/Services/MyTube/main.c . . . 99
src/Services/MyTube/thumbnailer.c . . . 297

src/Services/MyTube/thumbnailer.h 298
src/Services/MyURL/main.c 102
src/Services/ScriptRunner/main.cpp 298
src/Services/SimpleTemplate/main.c 106
src/Services/SQLiteServer/main.c 108
src/Services/SQLiteServer/sqlite.c 302
src/Services/SQLiteServer/sqlite.h 302
src/StringRecognizer/fastStringParser.c 303
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 305
src/StringRecognizer/main.c 109
src/UnitTests/testHashMap.c 307
src/UserAccounts/main.c 110
src/UserAccounts/userAccounts.h 308

Chapter 5

Data Structure Documentation

5.1 _list_item Struct Reference

```
#include <list.h>
```

Collaboration diagram for _list_item:

Data Fields

- struct _list_item ∗ next
- union {
  void ∗ item
  struct _list_item ∗ curr
  } ptr

5.1.1 Field Documentation

5.1.1.1 struct _list_item∗ curr

5.1.1.2 void∗ item

5.1.1.3 struct _list_item∗ next

5.1.1.4 union { ... } ptr

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

- src/Services/MyRemoteDesktop/xwd-1.0.5/list.h

5.2 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
- unsigned int contentContainsPathToFileToBeStreamed
- 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

5.2.1 Detailed Description

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

5.2.2 Field Documentation

5.2.2.1 unsigned int clientID

5.2.2.2 char∗ compressedContent

5.2.2.3 unsigned long compressedContentSize

5.2.2.4 char∗ content

5.2.2.5 unsigned int contentContainsPathToFileToBeStreamed

5.2.2.6 unsigned long contentSize

5.2.2.7 char∗ GET_request

5.2.2.8 unsigned int GET_request_length

5.2.2.9 unsigned int headerResponse

5.2.2.10 unsigned long MAXcompressedContentSize

5.2.2.11 unsigned long MAXcontentSize

5.2.2.12 char∗ POST_request

5.2.2.13 unsigned int POST_request_length

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

- src/AmmServerlib/AmmServerlib.h

5.3 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 int cacheVersionETag
- 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]

5.3.1 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.

5.3.2 Field Documentation

5.3.2.1 void∗ cache

5.3.2.2 void∗ cacheHashMap

5.3.2.3 unsigned int cacheVersionETag

5.3.2.4 unsigned int CLIENT_THREADS_STARTED

5.3.2.5 unsigned int CLIENT_THREADS_STOPPED

5.3.2.6 void∗ clientList

5.3.2.7 struct AmmServer_RequestOverride_Context∗ clientRequestHandlerOverrideContext

5.3.2.8 int files_open

5.3.2.9 char instanceName[MAX_INSTANCE_NAME_STRING]

5.3.2.10 unsigned int loaded_cache_items

5.3.2.11 unsigned long loaded_cache_items_Kbytes

5.3.2.12 int pause_server

5.3.2.13 unsigned int prespawn_jobs_finished

5.3.2.14 unsigned int prespawn_jobs_started

5.3.2.15 unsigned int prespawn_turn_to_serve

5.3.2.16 void* prespawned_pool

5.3.2.17 int server_running

5.3.2.18 pthread_t server_thread_id

5.3.2.19 int serversock

5.3.2.20 struct AmmServer_Instance_Settings settings

5.3.2.21 int stop_server

5.3.2.22 char templates_root[MAX_FILE_PATH]

5.3.2.23 pthread_t* threads_pool

5.3.2.24 char webserver_root[MAX_FILE_PATH]

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

- src/AmmServerlib/AmmServerlib.h

5.4 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

5.4.1 Detailed Description

Each Instance of AmmarServer has some basic settings , which are stored in AmmServer_Instance_Settings.

5.4.2 Field Documentation

5.4.2.1 char∗ BASE64PASSWORD

5.4.2.2 int BINDING_PORT

5.4.2.3 char∗ PASSWORD

5.4.2.4 int PASSWORD_PROTECTION

5.4.2.5 char∗ USERNAME

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

- src/AmmServerlib/AmmServerlib.h

5.5 AmmServer_MemoryHandler Struct Reference

A Wrapper around a memory buffer that enables house keeping for reallocations etc.

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

Data Fields

- unsigned int contentSize
- unsigned int contentCurrentLength
- char ∗ content

5.5.1 Detailed Description

A Wrapper around a memory buffer that enables house keeping for reallocations etc.

5.5.2 Field Documentation

5.5.2.1 char∗ content

5.5.2.2 unsigned int contentCurrentLength

5.5.2.3 unsigned int contentSize

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

- src/AmmServerlib/AmmServerlib.h

5.6 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

5.6.1 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.

5.6.2 Field Documentation

5.6.2.1 struct HTTPHeader∗ request

5.6.2.2 void∗ request_override_callback

5.6.2.3 char requestHeader[64]

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

- src/AmmServerlib/AmmServerlib.h

5.7 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 executedNow
- 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

5.7.1 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.

5.7.2 Field Documentation

5.7.2.1 char callback_cooldown

5.7.2.2 unsigned int callback_every_x_msec

5.7.2.3 void∗ dynamicRequestCallbackFunction

5.7.2.4 unsigned int executedNow

5.7.2.5 unsigned int last_callback

5.7.2.6 struct AmmServer_DynamicRequest requestContext

5.7.2.7 char resource_name[MAX_RESOURCE]

5.7.2.8 unsigned int RH_Scenario

5.7.2.9 char web_root_path[MAX_FILE_PATH]

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

- src/AmmServerlib/AmmServerlib.h

5.8 board Struct Reference

```
#include <state.h>
```

Collaboration diagram for board:

Data Fields

- char name [MAX_STRING_SIZE]
- unsigned int maxThreads
- unsigned int currentThreads
- unsigned int currentUsers
- unsigned int active
- unsigned int hidden
- unsigned int postUID
- unsigned int threadUID
- unsigned int imageUID
- unsigned int ∗ threadQueue
- struct thread ∗ threads

5.8.1 Field Documentation

5.8.1.1 unsigned int active

5.8.1.2 unsigned int currentThreads

5.8.1.3 unsigned int currentUsers

5.8.1.4 unsigned int hidden

5.8.1.5 unsigned int imageUID

5.8.1.6 unsigned int maxThreads

5.8.1.7 char name[MAX_STRING_SIZE]

5.8.1.8 unsigned int postUID

5.8.1.9 unsigned int∗ threadQueue

5.8.1.10 struct thread∗ threads

5.8.1.11 unsigned int threadUID

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

- src/Services/HabChan/state.h

5.9 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

5.9.1 Detailed Description

A cache item and all it's contents.

5.9.2 Field Documentation

5.9.2.1 char∗ compressedContent

5.9.2.2 unsigned long∗ compressedContentSize

5.9.2.3 char∗ content

5.9.2.4 unsigned long∗ contentSize

5.9.2.5 contentType contentTypeID

5.9.2.6 unsigned char doNOTCacheRule

5.9.2.7 struct AmmServer_RH_Context∗ dynamicRequest

5.9.2.8 void∗ dynamicRequestCallbackFunction

5.9.2.9 struct timestamp modification

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

- src/AmmServerlib/cache/file_caching.h

5.10 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

5.10.1 Detailed Description

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

5.10.2 Field Documentation

5.10.2.1 struct hashMap∗ userList

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

- src/AmmServerlib/cache/client_list.h

5.11 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

5.11.1 Detailed Description

Internal Structure that holds all the string parser context.

5.11.2 Field Documentation

5.11.2.1 struct **fspString**∗ contents

5.11.2.2 char∗ functionName

5.11.2.3 unsigned int longestStringLength

5.11.2.4 unsigned int MAXstringsLoaded

5.11.2.5 unsigned int shortestStringLength

5.11.2.6 unsigned int stringsLoaded

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

- src/StringRecognizer/fastStringParser.h

5.12 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

5.12.1 Detailed Description

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

5.12.2 Field Documentation

5.12.2.1 char∗ str

5.12.2.2 char∗ strIDFriendly

5.12.2.3 unsigned int strLength

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

- src/StringRecognizer/fastStringParser.h

5.13 guard_byte Struct Reference

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

Data Fields

- unsigned int checksum

5.13.1 Field Documentation

5.13.1.1 unsigned int checksum

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

- src/AmmServerlib/InputParser/InputParser_C.h

5.14 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

5.14.1 Detailed Description

The central structure for the hash map.

5.14.2 Field Documentation

5.14.2.1 void∗ clearItemCallbackFunction

5.14.2.2 unsigned int curNumberOfEntries

5.14.2.3 struct **hashMapEntry**∗ entries

5.14.2.4 unsigned int entryAllocationStep

5.14.2.5 pthread_mutex_t hm_addLock

5.14.2.6 pthread_mutex_t hm_fileLock

5.14.2.7 unsigned int maxNumberOfEntries

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

- src/AmmServerlib/hashmap/hashmap.h

5.15 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

5.15.1 Detailed Description

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

5.15.2 Field Documentation

5.15.2.1 unsigned int hits

5.15.2.2 char∗ key

5.15.2.3 unsigned long keyHash

5.15.2.4 unsigned int keyLength

5.15.2.5 void∗ payload

5.15.2.6 unsigned int payloadLength

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

- src/AmmServerlib/hashmap/hashmap.h

5.16 htmlContent Struct Reference

```
#include <database.h>
```

Data Fields

- unsigned int totalDataLength
- unsigned int currentDataLength
- unsigned char ∗ data

5.16.1 Field Documentation

5.16.1.1 unsigned int currentDataLength

5.16.1.2 unsigned char∗ data

5.16.1.3 unsigned int totalDataLength

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

- src/Services/MyBlog/database.h

5.17 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

5.17.1 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.

5.17.2 Field Documentation

5.17.2.1 unsigned char authorized

5.17.2.2 char∗ boundary

5.17.2.3 unsigned int boundaryLength

5.17.2.4 char∗ contentDisposition

5.17.2.5 unsigned int contentDispositionLength

5.17.2.6 unsigned long ContentLength

5.17.2.7 char∗ contentType

5.17.2.8 unsigned int contentTypeLength

5.17.2.9 char∗ cookie

5.17.2.10 unsigned int cookieLength

5.17.2.11 char∗ eTag

5.17.2.12 unsigned int eTagLength

5.17.2.13 char GETquery[MAX_QUERY+1]

5.17.2.14 char∗ headerRAW

5.17.2.15 unsigned int headerRAWSize

5.17.2.16 char∗ host

5.17.2.17 unsigned int hostLength

5.17.2.18 unsigned char keepalive

5.17.2.19 char∗ POSTrequest

5.17.2.20 unsigned long POSTrequestSize

5.17.2.21 unsigned long range_end

5.17.2.22 unsigned long range_start

5.17.2.23 char∗ referer

5.17.2.24 unsigned int refererLength

5.17.2.25 int requestType

5.17.2.26 char resource[MAX_RESOURCE+1]

5.17.2.27 unsigned char supports_compression

5.17.2.28 char∗ userAgent

5.17.2.29 unsigned int userAgentLength

5.17.2.30 char verified_local_resource[MAX_FILE_PATH+1]

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

- src/AmmServerlib/AmmServerlib.h

5.18 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

5.18.1 Detailed Description

Structure to keep data for an HTTP Transaction.

5.18.2 Field Documentation

5.18.2.1 unsigned int clientListID

5.18.2.2 int clientSock

5.18.2.3 struct HTTPHeader incomingHeader

5.18.2.4 struct AmmServer_Instance∗ instance

5.18.2.5 char∗ outgoingBody

5.18.2.6 unsigned int outgoingBodySize

5.18.2.7 int prespawnedThreadFlag

5.18.2.8 unsigned int resourceCacheID

5.18.2.9 unsigned int threadID

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

- src/AmmServerlib/AmmServerlib.h

5.19 Image Struct Reference

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

Data Fields

- unsigned char ∗ pixels
- unsigned int width
- unsigned int height
- unsigned int depth
- unsigned int imageSize

5.19.1 Field Documentation

5.19.1.1 unsigned int depth

5.19.1.2 unsigned int height

5.19.1.3 unsigned int imageSize

5.19.1.4 unsigned char∗ pixels

5.19.1.5 unsigned int width

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

- src/AmmCaptcha/imaging.h

5.20 image_region_type Struct Reference

Data Fields

- Window win
- Visual ∗ vis
- Colormap cmap
- int x_rootrel
- int y_rootrel
- int x_vis
- int y_vis
- int width
- int height
- int border
- Region visible_region

5.20.1 Field Documentation

5.20.1.1 int border

5.20.1.2 Colormap cmap

5.20.1.3 int height

5.20.1.4 Visual∗ vis

5.20.1.5 Region visible_region

5.20.1.6 int width

5.20.1.7 Window win

5.20.1.8 int x_rootrel

5.20.1.9 int x_vis

5.20.1.10 int y_rootrel

5.20.1.11 int y_vis

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

- src/Services/MyRemoteDesktop/xwd-1.0.5/multiVis.c

5.21 image_win_type Struct Reference

Data Fields

- Window win
- Visual ∗ vis
- Colormap cmap
- int x_rootrel
- int y_rootrel
- int x_vis
- int y_vis
- int width
- int height
- int border_width
- Window parent

5.21.1 Field Documentation

5.21.1.1 int border_width

5.21.1.2 Colormap cmap

5.21.1.3 int height

5.21.1.4 Window parent

5.21.1.5 Visual∗ vis

5.21.1.6 int width

5.21.1.7 Window win

5.21.1.8 int x_rootrel

5.21.1.9 int x_vis

5.21.1.10 int y_rootrel

5.21.1.11 int y_vis

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

- src/Services/MyRemoteDesktop/xwd-1.0.5/multiVis.c

5.22 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)

5.22.1 Constructor & Destructor Documentation

5.22.1.1 InputParser ( )

Here is the call graph for this function:

5.22.1.2 ∼InputParser ( )

Here is the call graph for this function:

5.22.2 Member Function Documentation

5.22.2.1 void DefaultDelimeterSetup ( )

Here is the call graph for this function:

5.22.2.2 char GetDelimeter (int *num*)

Here is the call graph for this function:

5.22.2.3 unsigned int GetLowercaseWord (int *num,* char ∗ *thestr,* unsigned int *thestrsize*)

Here is the call graph for this function:

5.22.2.4 unsigned int GetUpcaseWord (int *num,* char ∗ *thestr,* unsigned int *thestrsize*)

Here is the call graph for this function:

5.22.2.5 unsigned int GetWord (int *num,* char ∗ *thestr,* unsigned int *thestrsize*)

Here is the call graph for this function:

5.22.2.6 char GetWordChar (int *num,* int *chr*)

Here is the call graph for this function:

5.22.2.7 signed int GetWordInt (int *num*)

Here is the call graph for this function:

5.22.2.8 unsigned short GetWordLength (int *num*)

Here is the call graph for this function:

5.22.2.9 int SeperateWords (char ∗ *inpt*)

Here is the call graph for this function:

5.22.2.10 int SeperateWordsCC (const char ∗ *inpt*)

Here is the call graph for this function:

5.22.2.11 int SeperateWordsUC (unsigned char ∗ *inpt*)

Here is the call graph for this function:

5.22.2.12 void SetDelimeter (int *num,* char *tmp*)

Here is the call graph for this function:

5.22.2.13 const char∗ Version ( )

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

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

5.23 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

5.23.1 Field Documentation

5.23.1.1 char∗ container_end

5.23.1.2 char∗ container_start

5.23.1.3 unsigned short cur_container_count

5.23.1.4 unsigned short cur_delimeter_count

5.23.1.5 char∗ delimeters

5.23.1.6 struct **guard_byte** guardbyte1

5.23.1.7 struct **guard_byte** guardbyte2

5.23.1.8 struct **guard_byte** guardbyte3

5.23.1.9 struct **guard_byte** guardbyte4

5.23.1.10 unsigned char local_allocation

5.23.1.11 unsigned short max_container_count

5.23.1.12 unsigned short max_delimeter_count

5.23.1.13 char∗ str

5.23.1.14 unsigned int str_length

5.23.1.15 struct **tokens**∗ tokenlist

5.23.1.16 unsigned int tokens_count

5.23.1.17 unsigned int tokens_max

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

- src/AmmServerlib/InputParser/InputParser_C.h

5.24 linkItemList Struct Reference

```
#include <database.h>
```

Collaboration diagram for linkItemList:

Data Fields

- unsigned int currentItems
- unsigned int maxItems
- struct linkLabelItem item [MAX_MENU_ITEMS]

5.24.1 Field Documentation

5.24.1.1 unsigned int currentItems

5.24.1.2 struct **linkLabelItem** item[**MAX_MENU_ITEMS**]

5.24.1.3 unsigned int maxItems

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

- src/Services/MyBlog/database.h

5.25 linkLabelItem Struct Reference

```
#include <database.h>
```

Data Fields

- unsigned char label [MAX_STR]
- unsigned char link [MAX_STR]

5.25.1 Field Documentation

5.25.1.1 unsigned char label[MAX_STR]

5.25.1.2 unsigned char link[MAX_STR]

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

- src/Services/MyBlog/database.h

5.26 menuItemList Struct Reference

`#include <database.h>`

Collaboration diagram for menuItemList:

Data Fields

- unsigned int currentItems
- unsigned int maxItems
- struct linkLabelItem item [MAX_MENU_ITEMS]

5.26.1 Field Documentation

5.26.1.1 unsigned int currentItems

5.26.1.2 struct linkLabelItem item[MAX_MENU_ITEMS]

5.26.1.3 unsigned int maxItems

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

- src/Services/MyBlog/database.h

5.27 my_XRegion Struct Reference

Collaboration diagram for my_XRegion:

Data Fields

- long size
- long numRects
- myBOX ∗ rects
- myBOX extents

5.27.1 Field Documentation

5.27.1.1 myBOX extents

5.27.1.2 long numRects

5.27.1.3 myBOX∗ rects

5.27.1.4 long size

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

- src/Services/MyRemoteDesktop/xwd-1.0.5/multiVis.c

5.28 myBox Struct Reference

Data Fields

- short x1
- short x2
- short y1
- short y2

5.28.1 Detailed Description

This file contains functions to create a list of regions which tile a specified window. Each region contains all visible portions of the window which are drawn with the same visual. If the window consists of subwindows of two different visual types, there will be two regions in the list. The list can be traversed to correctly pull an image of the window using XGetImage or the Image Library.

Copyright 1994 Hewlett-Packard Co. Copyright 1996, 1998 The Open Group

Permission to use, copy, modify, distribute, and sell this software and its documentation for any purpose is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE OPEN GROUP BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of The Open Group shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from The Open Group.

5.28.2 Field Documentation

5.28.2.1 short x1

5.28.2.2 short x2

5.28.2.3 short y1

5.28.2.4 short y2

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

- src/Services/MyRemoteDesktop/xwd-1.0.5/multiVis.c

5.29 OverlayInfo Struct Reference

```
#include <wsutils.h>
```

Data Fields

- XVisualInfo ∗ pOverlayVisualInfo
- int transparentType
- Pixel value
- int layer

5.29.1 Field Documentation

5.29.1.1 int layer

5.29.1.2 XVisualInfo∗ pOverlayVisualInfo

5.29.1.3 int transparentType

5.29.1.4 Pixel value

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

- src/Services/MyRemoteDesktop/xwd-1.0.5/wsutils.h

5.30 OverlayVisualPropertyRec Struct Reference

```
#include <wsutils.h>
```

Data Fields

- VisualID visualID
- int transparentType
- Pixel value
- int layer

5.30.1 Detailed Description

This file contains routines for manipulating generic lists. Lists are implemented with a "harness". In other words, each node in the list consists of two pointers, one to the data item and one to the next node in the list. The head of the list is the same struct as each node, but the "item" ptr is used to point to the current member of the list (used by the first_in_list and next_in_list functions).

5.30.2 Field Documentation

5.30.2.1 int layer

5.30.2.2 int transparentType

5.30.2.3 Pixel value

5.30.2.4 VisualID visualID

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

- src/Services/MyRemoteDesktop/xwd-1.0.5/wsutils.h

5.31 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
- char ip [MAX_IP_STRING_SIZE]

5.31.1 Detailed Description

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

5.31.2 Field Documentation

5.31.2.1 struct sockaddr_in client

5.31.2.2 unsigned int clientlen

5.31.2.3 int clientsock

5.31.2.4 struct **AmmServer_Instance**∗ instance

5.31.2.5 char ip[**MAX_IP_STRING_SIZE**]

5.31.2.6 volatile int keep_var_on_stack

5.31.2.7 unsigned int port

5.31.2.8 int pre_spawned_thread

5.31.2.9 unsigned int thread_id

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

- src/AmmServerlib/threads/freshThreads.h

5.32 PassToPreSpawnedThread Struct Reference

Collaboration diagram for PassToPreSpawnedThread:

Data Fields

- struct AmmServer_Instance ∗ instance
- unsigned int i_adapt

5.32.1 Field Documentation

5.32.1.1 unsigned int i_adapt

5.32.1.2 struct **AmmServer_Instance**∗ instance

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

- src/AmmServerlib/threads/prespawnedThreads.c

5.33 playlist Struct Reference

Collaboration diagram for playlist:

Data Fields

- unsigned int numberOfItems
- unsigned int maxItems
- struct playlistItem item [100]
- unsigned int playlistActiveItem
- unsigned int playlistState

5.33.1 Field Documentation

5.33.1.1 struct **playlistItem** item[100]

5.33.1.2 unsigned int maxItems

5.33.1.3 unsigned int numberOfItems

5.33.1.4 unsigned int playlistActiveItem

5.33.1.5 unsigned int playlistState

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

- src/Services/CinemaPilot/main.c

5.34 playlistItem Struct Reference

Data Fields

- int command
- char playFile [512]
- struct tm ∗ triggerTime
- struct tm ∗ stopTime

5.34.1 Field Documentation

5.34.1.1 int command

5.34.1.2 char playFile[512]

5.34.1.3 struct tm∗ stopTime

5.34.1.4 struct tm∗ triggerTime

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

- src/Services/CinemaPilot/main.c

5.35 post Struct Reference

```
#include <state.h>
```

Collaboration diagram for post:

Data Fields

- unsigned int numberOfComplaints
- char op [MAX_STRING_SIZE]
- char password [MAX_STRING_SIZE]
- unsigned char hasFile
- unsigned char fileType

- char fileOriginalName [MAX_STRING_SIZE]
- char fileCachedName [MAX_STRING_SIZE]
- unsigned int fileDimensionWidth
- unsigned int fileDimensionHeight
- struct timestamp creation
- unsigned int messageSize
- char ∗ message

5.35.1 Field Documentation

5.35.1.1 struct timestamp creation

5.35.1.2 char fileCachedName[MAX_STRING_SIZE]

5.35.1.3 unsigned int fileDimensionHeight

5.35.1.4 unsigned int fileDimensionWidth

5.35.1.5 char fileOriginalName[MAX_STRING_SIZE]

5.35.1.6 unsigned char fileType

5.35.1.7 unsigned char hasFile

5.35.1.8 char∗ message

5.35.1.9 unsigned int messageSize

5.35.1.10 unsigned int numberOfComplaints

5.35.1.11 char op[MAX_STRING_SIZE]

5.35.1.12 char password[MAX_STRING_SIZE]

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

- src/Services/HabChan/state.h

5.36 postItem Struct Reference

```
#include <database.h>
```

Collaboration diagram for postItem:

Data Fields

- unsigned char title [MAX_STR]
- unsigned char dateStr [MAX_STR]
- unsigned char author [MAX_STR]
- struct tagItemList tags
- struct htmlContent content

5.36.1 Field Documentation

5.36.1.1 unsigned char author[MAX_STR]

5.36.1.2 struct htmlContent content

5.36.1.3 unsigned char dateStr[MAX_STR]

5.36.1.4 struct tagItemList tags

5.36.1.5 unsigned char title[MAX_STR]

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

- src/Services/MyBlog/database.h

5.37 postItemList Struct Reference

```
#include <database.h>
```

Collaboration diagram for postItemList:

Data Fields

- unsigned int currentPosts
- unsigned int maxPosts
- struct postItem item [MAX_TAGS_PER_POST]

5.37.1 Field Documentation

5.37.1.1 unsigned int currentPosts

5.37.1.2 struct postItem item[MAX_TAGS_PER_POST]

5.37.1.3 unsigned int maxPosts

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

- src/Services/MyBlog/database.h

5.38 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]

5.38.1 Detailed Description

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

5.38.2 Field Documentation

5.38.2.1 volatile unsigned char busy

5.38.2.2 struct sockaddr_in client

5.38.2.3 unsigned int clientlen

5.38.2.4 int clientsock

5.38.2.5 struct **AmmServer_Instance**∗ instance

5.38.2.6 char templates_root[**MAX_FILE_PATH**]

5.38.2.7 pthread_t thread_id

5.38.2.8 unsigned int threadNum

5.38.2.9 char webserver_root[**MAX_FILE_PATH**]

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

- src/AmmServerlib/threads/prespawnedThreads.h

5.39 site Struct Reference

```
#include <state.h>
```

Collaboration diagram for site:

Data Fields

- unsigned int maxNumberOfBoards
- unsigned int numberOfBoards
- struct board ∗ boards
- char siteName [MAX_STRING_SIZE]
- char siteDescription [MAX_STRING_SIZE]

5.39.1 Field Documentation

5.39.1.1 struct board∗ boards

5.39.1.2 unsigned int maxNumberOfBoards

5.39.1.3 unsigned int numberOfBoards

5.39.1.4 char siteDescription[MAX_STRING_SIZE]

5.39.1.5 char siteName[MAX_STRING_SIZE]

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

- src/Services/HabChan/state.h

5.40 socialLinks Struct Reference

```
#include <database.h>
```

Data Fields

- unsigned char facebookURL [MAX_STR]
- unsigned char twitterURL [MAX_STR]
- unsigned char youtubeURL [MAX_STR]

5.40.1 Field Documentation

5.40.1.1 unsigned char facebookURL[MAX_STR]

5.40.1.2 unsigned char twitterURL[MAX_STR]

5.40.1.3 unsigned char youtubeURL[MAX_STR]

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

- src/Services/MyBlog/database.h

5.41 SQLiteSession Struct Reference

```
#include <database.h>
```

Data Fields

- sqlite3 ∗ db
- sqlite3_stmt ∗ res
- char ∗ err_msg
- int rc

5.41.1 Field Documentation

5.41.1.1 sqlite3 ∗ db

5.41.1.2 char ∗ err_msg

5.41.1.3 int rc

5.41.1.4 sqlite3_stmt ∗ res

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

- src/Services/MyBlog/database.h
- src/Services/MyBlog/tools/myblogTool.c
- src/Services/SQLiteServer/sqlite.h

5.42 tagItem Struct Reference

```
#include <database.h>
```

Data Fields

- unsigned char tag [MAX_STR]
- unsigned int tagHash

5.42.1 Field Documentation

5.42.1.1 unsigned char tag[MAX_STR]

5.42.1.2 unsigned int tagHash

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

- src/Services/MyBlog/database.h

5.43 tagItemList Struct Reference

```
#include <database.h>
```

Collaboration diagram for tagItemList:

Data Fields

- unsigned int currentTags
- unsigned int maxTags
- struct tagItem item [MAX_TAGS_PER_POST]

5.43.1 Field Documentation

5.43.1.1 unsigned int currentTags

5.43.1.2 struct tagItem item[MAX_TAGS_PER_POST]

5.43.1.3 unsigned int maxTags

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

- src/Services/MyBlog/database.h

5.44 thread Struct Reference

```
#include <state.h>
```

Collaboration diagram for thread:

Data Fields

- unsigned char sticky
- unsigned char repliable
- char op [MAX_STRING_SIZE]
- char password [MAX_STRING_SIZE]
- char title [MAX_STRING_SIZE]
- struct timestamp creation
- struct timestamp lastReply
- unsigned int maxNumberOfReplies
- unsigned int numberOfReplies
- unsigned int numberOfImages
- struct post ∗ replies

5.44.1 Field Documentation

5.44.1.1 struct timestamp creation

5.44.1.2 struct timestamp lastReply

5.44.1.3 unsigned int maxNumberOfReplies

5.44.1.4 unsigned int numberOfImages

5.44.1.5 unsigned int numberOfReplies

5.44.1.6 char op[MAX_STRING_SIZE]

5.44.1.7 char password[MAX_STRING_SIZE]

5.44.1.8 unsigned char repliable

5.44.1.9 struct post∗ replies

5.44.1.10 unsigned char sticky

5.44.1.11 char title[MAX_STRING_SIZE]

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

- src/Services/HabChan/state.h

5.45 time_snap Struct Reference

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

Data Fields

- struct timeval starttime endtime difference

5.45.1 Field Documentation

5.45.1.1 struct timeval starttime endtime difference

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

- src/AmmServerlib/tools/time_provider.h

5.46 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
- unsigned int day
- unsigned int month
- unsigned int hour
- unsigned int minute
- unsigned int second

5.46.1 Detailed Description

Timestamp for a cache item entry.

5.46.2 Field Documentation

5.46.2.1 unsigned char day

5.46.2.2 unsigned int day

5.46.2.3 unsigned char hour

5.46.2.4 unsigned int hour

5.46.2.5 unsigned char minute

5.46.2.6 unsigned int minute

5.46.2.7 unsigned char month

5.46.2.8 unsigned int month

5.46.2.9 unsigned char second

5.46.2.10 unsigned int second

5.46.2.11 unsigned char wday

5.46.2.12 unsigned int year

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

- src/AmmServerlib/cache/file_caching.h
- src/Services/HabChan/state.h

5.47 tokens Struct Reference

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

Data Fields

- unsigned int token_start
- unsigned int length

5.47.1 Field Documentation

5.47.1.1 unsigned int length

5.47.1.2 unsigned int token_start

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

- src/AmmServerlib/InputParser/InputParser_C.h

5.48 URLDB Struct Reference

Data Fields

- char ∗ longURL
- char ∗ shortURL
- unsigned long shortURLHash

5.48.1 Field Documentation

5.48.1.1 char∗ longURL

5.48.1.2 char∗ shortURL

5.48.1.3 unsigned long shortURLHash

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

- src/Services/MyURL/main.c

5.49 UserAccountAuthenticationToken Struct Reference

```
#include <userAccounts.h>
```

Data Fields

- unsigned int dummy

5.49.1 Field Documentation

5.49.1.1 unsigned int dummy

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

- src/UserAccounts/userAccounts.h

5.50 UserAccountDatabase Struct Reference

```
#include <userAccounts.h>
```

Collaboration diagram for UserAccountDatabase:

Data Fields

- unsigned int dummy
- struct
  UserAccountAuthenticationToken lastAuthenticationToken

5.50.1 Field Documentation

5.50.1.1 unsigned int dummy

5.50.1.2 struct UserAccountAuthenticationToken lastAuthenticationToken

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

- src/UserAccounts/userAccounts.h

5.51 videoCollection Struct Reference

```
#include <indexer.h>
```

Collaboration diagram for videoCollection:

Data Fields

- struct videoItem ∗ video
- unsigned int numberOfLoadedVideos
- unsigned int MAX_numberOfVideos

5.51.1 Field Documentation

5.51.1.1 unsigned int MAX_numberOfVideos

5.51.1.2 unsigned int numberOfLoadedVideos

5.51.1.3 struct videoItem∗ video

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

- src/Services/MyTube/indexer.h

5.52 videoItem Struct Reference

```
#include <indexer.h>
```

Data Fields

- unsigned long hashID
- unsigned long views
- unsigned long likes
- unsigned long dislikes
- unsigned int visibility
- char title [MAX_STR]
- char tagsStr [MAX_STR]
- char filename [MAX_STR]
- char comment [MAX_STR]
- char thumbnail [MAX_STR]

5.52.1 Field Documentation

5.52.1.1 char comment[MAX_STR]

5.52.1.2 unsigned long dislikes

5.52.1.3 char filename[MAX_STR]

5.52.1.4 unsigned long hashID

5.52.1.5 unsigned long likes

5.52.1.6 char tagsStr[MAX_STR]

5.52.1.7 char thumbnail[MAX_STR]

5.52.1.8 char title[MAX_STR]

5.52.1.9 unsigned long views

5.52.1.10 unsigned int visibility

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

- src/Services/MyTube/indexer.h

5.53 website Struct Reference

```
#include <database.h>
```

Collaboration diagram for website:

Data Fields

- int allowComments
- int allowPing
- unsigned char blogTitle [MAX_STR]
- unsigned char siteName [MAX_STR]
- unsigned char siteDescription [MAX_STR]
- unsigned char siteURL [MAX_STR]
- struct socialLinks social
- struct menuItemList menu
- struct linkItemList linksLeft
- struct linkItemList linksRight
- struct postItemList post
- struct widgetItemList widget

5.53.1 Field Documentation

5.53.1.1 int allowComments

5.53.1.2 int allowPing

5.53.1.3 unsigned char blogTitle[MAX_STR]

5.53.1.4 struct linkItemList linksLeft

5.53.1.5 struct linkItemList linksRight

5.53.1.6 struct menuItemList menu

5.53.1.7 struct postItemList post

5.53.1.8 unsigned char siteDescription[MAX_STR]

5.53.1.9 unsigned char siteName[MAX_STR]

5.53.1.10 unsigned char siteURL[MAX_STR]

5.53.1.11 struct socialLinks social

5.53.1.12 struct widgetItemList widget

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

- src/Services/MyBlog/database.h

5.54 widgetItem Struct Reference

`#include <database.h>`

Collaboration diagram for widgetItem:

Data Fields

- unsigned char label [MAX_STR]
- unsigned char link [MAX_STR]
- struct htmlContent content

5.54.1 Field Documentation

5.54.1.1 struct htmlContent content

5.54.1.2 unsigned char label[MAX_STR]

5.54.1.3 unsigned char link[MAX_STR]

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

- src/Services/MyBlog/database.h

5.55 widgetItemList Struct Reference

`#include <database.h>`

Collaboration diagram for widgetItemList:

Data Fields

- unsigned int currentItems
- unsigned int maxItems
- struct widgetItem item [MAX_WIDGET_ITEMS]

5.55.1 Field Documentation

5.55.1.1 unsigned int currentItems

5.55.1.2 struct **widgetItem** item[**MAX_WIDGET_ITEMS**]

5.55.1.3 unsigned int maxItems

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

- src/Services/MyBlog/database.h

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 ----------------------—

Here is the call graph for this function:

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 captchaID, char ∗reply)
- int AmmCaptcha_getCaptchaFrame (unsigned int captchaID, char ∗mem, unsigned long ∗mem_size)
- int AmmCaptcha_getJPEGFileFromPixels (char ∗pixels, unsigned int width, unsigned int height, unsigned int channels, char ∗mem, unsigned long ∗mem_size)
- int testAmmCaptcha ()

6.3.1 Function Documentation

6.3.1.1 int AmmCaptcha_destroy ( )

Here is the call graph for this function:

6.3.1.2 int AmmCaptcha_getCaptchaFrame (unsigned int *captchaID,* char ∗ *mem,* unsigned long ∗ *mem_size*)

Here is the call graph for this function:

6.3.1.3 int AmmCaptcha_getJPEGFileFromPixels (char ∗ *pixels,* unsigned int *width,* unsigned int *height,* unsigned int *channels,* char ∗ *mem,* unsigned long ∗ *mem_size*)

Here is the call graph for this function:

6.3.1.4 int AmmCaptcha_initialize (char ∗ *font,* char ∗ *dictFilename*)

Here is the call graph for this function:

6.3.1.5 int AmmCaptcha_isReplyCorrect (unsigned int *captchaID,* char ∗ *reply*)

Here is the call graph for this function:

6.3.1.6 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 captchaID)
- int AmmCaptcha_isReplyCorrect (unsigned int captchaID, char ∗reply)
- int AmmCaptcha_getCaptchaFrame (unsigned int captchaID, char ∗mem, unsigned long ∗mem_size)
- int AmmCaptcha_getJPEGFileFromPixels (char ∗pixels, unsigned int width, unsigned int height, unsigned int channels, 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 *captchaID,* char ∗ *mem,* unsigned long ∗ *mem_size*)

Here is the call graph for this function:

6.5.2.2 int AmmCaptcha_destroy ( )

Here is the call graph for this function:

6.5.2.3 int AmmCaptcha_getCaptchaFrame (unsigned int *captchaID,* char ∗ *mem,* unsigned long ∗ *mem_size*)

Here is the call graph for this function:

6.5.2.4 int AmmCaptcha_getJPEGFileFromPixels (char ∗ *pixels,* unsigned int *width,* unsigned int *height,* unsigned int *channels,* char ∗ *mem,* unsigned long ∗ *mem_size*)

Here is the call graph for this function:

6.5.2.5 int AmmCaptcha_initialize (char ∗ *font,* char ∗ *dictFilename*)

Here is the call graph for this function:

6.5.2.6 int AmmCaptcha_isReplyCorrect (unsigned int *captchaID,* char ∗ *reply*)

Here is the call graph for this function:

6.5.2.7 int AmmCaptcha_loadDictionary (char ∗ *dictFilename*)

Here is the call graph for this function:

6.5.2.8 unsigned int convertExternalIDToInternal (unsigned int *captchaID*)

Here is the call graph for this function:

6.5.2.9 int RenderString (struct Image ∗ *frame*, struct Image ∗ *font*, unsigned int *x*, unsigned int *y*, char ∗ *str*)

Here is the call graph for this function:

6.5.2.10 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:

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 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 BLACK "\033[30m" /∗ Black ∗/

6.6.1.2 #define BLUE "\033[34m" /∗ Blue ∗/

6.6.1.3 #define CYAN "\033[36m" /∗ Cyan ∗/

6.6.1.4 #define GREEN "\033[32m" /∗ Green ∗/

6.6.1.5 #define MAGENTA "\033[35m" /∗ Magenta ∗/

6.6.1.6 #define max_ret_word 256

6.6.1.7 #define NORMAL "\033[0m"

6.6.1.8 #define RED "\033[31m" /∗ Red ∗/

6.6.1.9 #define WHITE "\033[37m" /∗ White ∗/

6.6.1.10 #define YELLOW "\033[33m" /∗ Yellow ∗/

6.6.2 Function Documentation

6.6.2.1 int IntermediateTests ( )

Here is the call graph for this function:

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*)

Here is the call graph for this function:

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 "AString/AString.h"
#include "threads/threadedServer.h"
#include "threads/prespawnedThreads.h"
#include "cache/file_caching.h"
#include "cache/dynamic_requests.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 (const char ∗name, const char ∗ip, unsigned int port, const char ∗conf_file, const char ∗web_root_path, const char ∗templates_root_path)

  Start a Web Server , allocate memory , bind ports and return its instance..
- struct AmmServer_Instance ∗ AmmServer_StartWithArgs (const char ∗name, int argc, char ∗∗argv, const char ∗ip, unsigned int port, const char ∗conf_file, const char ∗web_root_path, const 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_DynamicRequestReturnFile (struct AmmServer_DynamicRequest ∗rqst, const char ∗filename)

  Return a file instead of a Dynamic Request.
- int AmmServer_AddRequestHandler (struct AmmServer_Instance ∗instance, struct AmmServer_RequestOverride_Context ∗RequestOverrideContext, const 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, const char ∗resource_name, const 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, const 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, const 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, const 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, const 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, const 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, const 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, const char ∗var_id_IN, 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, const 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 (const 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, const 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 (const 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.
- void AmmServer_ReplaceCharInString (char ∗input, char findChar, char replaceWith)

  Hot-Replace a character inside a memory block , typically used to replace characters like '+' with ' '.
- int AmmServer_ReplaceVarInMemoryFile (char ∗page, unsigned int pageLength, const char ∗var, const 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.

- int AmmServer_ReplaceAllVarsInMemoryFile (char ∗page, unsigned int instances, unsigned int pageLength, const char ∗var, const char ∗value)

    Hot-Replace ALL variables inside a memory block , typically used to replace placeholders inside text files , like $$$$$$$$NAME$$$$$$$$, the value should be smaller or equal to the var being 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_ExecuteCommandLineNum (const char ∗command, char ∗what2GetBack, unsigned int what2GetBackMaxSize, unsigned int lineNumber)

    Execute a command and copy its output line to the provided buffer.
- int AmmServer_ExecuteCommandLine (const char ∗command, char ∗what2GetBack, unsigned int what2GetBackMaxSize)

    Execute a command and copy its output to the provided buffer.
- char ∗ AmmServer_ReadFileToMemory (const char ∗filename, unsigned int ∗length)

    Read a file and store it to a freshly allocated memory block.
- int AmmServer_WriteFileFromMemory (const char ∗filename, char ∗memory, unsigned int memoryLength)

    Dump a memory block to a file.
- int AmmServer_CopyOverlappingDataContent (unsigned char ∗buffer, unsigned int totalSize, unsigned char ∗from, unsigned char ∗to, unsigned int blockSize)

    Copy Content from one place of a buffer to another using an intermediate buffer..
- int AmmServer_InjectDataToBuffer (unsigned char ∗entryPoint, unsigned char ∗data, struct AmmServer_MemoryHandler ∗mh)

    Search for entryPoint pattern in buffer , and inject data there..!
- int AmmServer_ReplaceVarInMemoryHandler (struct AmmServer_MemoryHandler ∗mh, const char ∗var, const char ∗value)
- int AmmServer_ReplaceAllVarsInMemoryHandler (struct AmmServer_MemoryHandler ∗mh, unsigned int instances, const char ∗var, const char ∗value)
- struct AmmServer_MemoryHandler ∗ AmmServer_AllocateMemoryHandler (unsigned int initialBufferLength, unsigned int growStep)
- struct AmmServer_MemoryHandler ∗ AmmServer_ReadFileToMemoryHandler (const char ∗filename)

    Read a file and store it to a freshly allocated memory handler context.
- struct AmmServer_MemoryHandler ∗ AmmServer_CopyMemoryHandler (struct AmmServer_MemoryHandler ∗inpt)

    Copy a memory handler.
- int AmmServer_FreeMemoryHandler (struct AmmServer_MemoryHandler ∗∗mh)
- int AmmServer_ConvertBufferToMemoryHandler (struct AmmServer_MemoryHandler ∗mh, unsigned char ∗buffer, unsigned int bufferLength)
- int AmmServer_DirectoryExists (const char ∗filename)

    Check if directory Exists.
- int AmmServer_FileExists (const char ∗filename)

    Check if file Exists.
- int AmmServer_FileIsVideo (const char ∗filename)

    Check if file is a video.
- int AmmServer_EraseFile (const char ∗filename)

    Erase a File.
- unsigned int AmmServer_StringIsHTMLSafe (const 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,* const 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,* const 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,* const 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,* const 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
A	AmmServer_RequestOverride_Context to be populated
Request	Type
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,* const char ∗ *resource_name,* const 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 struct **AmmServer_MemoryHandler*** AmmServer_AllocateMemoryHandler (unsigned int *initialBufferLength*, unsigned int *growStep*)

6.7.1.7 int AmmServer_CheckIfHeaderBinaryAreTheSame (int *headerSpec*)

Internal Check to compare against changes of the header files.

Parameters

Header	(should be AMMAR_SERVER_HTTP_HEADER_SPEC)

Return values

1=Success,0=Failure	

6.7.1.8 int AmmServer_ConvertBufferToMemoryHandler (struct **AmmServer_MemoryHandler** * *mh*, unsigned char * *buffer*, unsigned int *bufferLength*)

6.7.1.9 struct **AmmServer_MemoryHandler*** AmmServer_CopyMemoryHandler (struct **AmmServer_MemoryHandler** * *inpt*)

Copy a memory handler.

Parameters

Input	memory handle

Return values

Pointer	to the new memory handler or 0=Failed

6.7.1.10 int AmmServer_CopyOverlappingDataContent (unsigned char * *buffer*, unsigned int *totalSize*, unsigned char * *from*, unsigned char * *to*, unsigned int *blockSize*)

Copy Content from one place of a buffer to another using an intermediate buffer..

Parameters

Original	Buffer
Size	of Original Buffer

Pointer	to the start of the source of the copy
Pointer	to the start of the destination of the copy
Size	of data to copy

Return values

1=Ok,0=Failed	

Here is the call graph for this function:

6.7.1.11 int AmmServer_DirectoryExists (const 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:

6.7.1.12 int AmmServer_DoNOTCacheResource (struct AmmServer_Instance ∗ *instance,* const 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:

6.7.1.13 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:

6.7.1.14 int AmmServer_DynamicRequestReturnFile (struct AmmServer_DynamicRequest ∗ *rqst,* const char ∗ *filename*)

Return a file instead of a Dynamic Request.

Parameters

An	AmmarServer Request
File	to serve

Return values

1=Running,0=Stopped	

6.7.1.15 int AmmServer_EraseFile (const char ∗ *filename*)

Erase a File.

Parameters

Path	to file

Return values

1=Success,0=Failure	

6.7.1.16 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	printf (`http://www.cplusplus.com/reference/cstdio/printf/`)
Arbitrary	number of other parameters that where defined in format

Here is the call graph for this function:

6.7.1.17 int AmmServer_ExecuteCommandLine (const 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

Here is the call graph for this function:

6.7.1.18 int AmmServer_ExecuteCommandLineNum (const char ∗ *command,* char ∗ *what2GetBack,* unsigned int *what2GetBackMaxSize,* unsigned int *lineNumber*)

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

Parameters

Command	to execute
Allocated	memory to store the result
Size	of Allocated memory
Number	of line we want to get back

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

6.7.1.19 int AmmServer_FileExists (const 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:

6.7.1.20 int AmmServer_FileIsVideo (const char ∗ *filename*)

Check if file is a video.

Parameters

Path	to file

Return values

1=Exists,0=Does	not Exist

Here is the call graph for this function:

6.7.1.21 int AmmServer_FILES (struct AmmServer_Instance ∗ *instance,* struct AmmServer_DynamicRequest ∗ *rqst,* const 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	

6.7.1.22 int AmmServer_FreeMemoryHandler (struct **AmmServer_MemoryHandler** ∗∗ *mh*)

6.7.1.23 void AmmServer_GeneralPrint (char ∗ *color,* char ∗ *label,* const char ∗ *format,* va_list ∗ *arglist*)

6.7.1.24 int AmmServer_GETArg (struct **AmmServer_Instance** ∗ *instance,* struct **AmmServer_DynamicRequest** ∗ *rqst,* const 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:

6.7.1.25 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

6.7.1.26 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

6.7.1.27 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

6.7.1.28 void AmmServer_GlobalTerminationHandler (int *signum*)

6.7.1.29 int AmmServer_InjectDataToBuffer (unsigned char ∗ *entryPoint,* unsigned char ∗ *data,* struct **AmmServer_MemoryHandler** ∗ *mh*)

Search for entryPoint pattern in buffer , and inject data there..!

Parameters

String	to find in buffer and replace with new content
Data	we want to inject
Memory	Handler for Buffer we want to inject to , see struct AmmServer_MemoryHandler

Return values

1=Ok,0=Failed	

Here is the call graph for this function:

6.7.1.30 int AmmServer_POSTArg (struct **AmmServer_Instance** ∗ *instance,* struct **AmmServer_DynamicRequest** ∗ *rqst,* const 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:

6.7.1.31 int AmmServer_PreCacheFile (struct **AmmServer_Instance** ∗ *instance,* const char ∗ *filename*)

Here is the call graph for this function:

6.7.1.32 char∗ AmmServer_ReadFileToMemory (const 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:

6.7.1.33 struct AmmServer_MemoryHandler∗ AmmServer_ReadFileToMemoryHandler (const char ∗ *filename*)

Read a file and store it to a freshly allocated memory handler context.

Parameters

Input	Filename

Return values

Pointer	to the new memory handler or 0=Failed

Here is the call graph for this function:

6.7.1.34 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:

6.7.1.35 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:

6.7.1.36 int AmmServer_ReplaceAllVarsInMemoryFile (char ∗ *page,* unsigned int *instances,* unsigned int *pageLength,* const char ∗ *var,* const char ∗ *value*)

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

Parameters

Pointer	to memory that contains the document
Maximum	number of Variable instances , 0 means infinite (until the end of the memory buffer)..
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 , this should be handled

Here is the call graph for this function:

6.7.1.37 int AmmServer_ReplaceAllVarsInMemoryHandler (struct **AmmServer_MemoryHandler** ∗ *mh,* unsigned int *instances,* const char ∗ *var,* const char ∗ *value*)

Here is the call graph for this function:

6.7.1.38 void AmmServer_ReplaceCharInString (char ∗ *input,* char *findChar,* char *replaceWith*)

Hot-Replace a character inside a memory block , typically used to replace characters like '+' with ' '.

Parameters

Pointer	to memory that contains the null terminated string
Character	to be replaced
What	to replace the character with

6.7.1.39 int AmmServer_ReplaceVarInMemoryFile (char ∗ *page,* unsigned int *pageLength,* const char ∗ *var,* const 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 , this should be handled

Here is the call graph for this function:

6.7.1.40 int AmmServer_ReplaceVarInMemoryHandler (struct **AmmServer_MemoryHandler** ∗ *mh,* const char ∗ *var,* const char ∗ *value*)

Here is the call graph for this function:

6.7.1.41 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:

6.7.1.42 int AmmServer_SaveDynamicRequest (const 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:

6.7.1.43 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

6.7.1.44 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

Here is the call graph for this function:

6.7.1.45 int AmmServer_SetStrSettingValue (struct **AmmServer_Instance** ∗ *instance,* unsigned int *set_type,* const 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:

6.7.1.46 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

6.7.1.47 struct **AmmServer_Instance**∗ AmmServer_Start (const char ∗ *name,* const char ∗ *ip,* unsigned int *port,* const char ∗ *conf_file,* const char ∗ *web_root_path,* const 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:

6.7.1.48 struct **AmmServer_Instance**∗ AmmServer_StartAdminInstance (const 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

6.7.1.49 struct **AmmServer_Instance**∗ AmmServer_StartWithArgs (const char ∗ *name,* int *argc,* char ∗∗ *argv,* const char ∗ *ip,* unsigned int *port,* const char ∗ *conf_file,* const char ∗ *web_root_path,* const 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:

6.7.1.50 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:

6.7.1.51 unsigned int AmmServer_StringIsHTMLSafe (const 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

1=Safe,0=Unsafe	

6.7.1.52 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	printf (`http://www.cplusplus.com/reference/cstdio/printf/`)
Arbitrary	number of other parameters that where defined in format

Here is the call graph for this function:

6.7.1.53 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.

6.7.1.54 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	printf (`http://www.cplusplus.com/reference/cstdio/printf/`)
Arbitrary	number of other parameters that where defined in format

Here is the call graph for this function:

6.7.1.55 int AmmServer_WriteFileFromMemory (const 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	

Here is the call graph for this function:

6.7.2 Variable Documentation

6.7.2.1 void(∗ TerminationCallback)()=0

6.8 src/ScriptRunner/main.c 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.c:

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 ∗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
- 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}

6.8.1 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_COMMAND_SIZE 2048

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 execute (char ∗ *command,* char ∗ *param*)

bin/bash -c "

Here is the call graph for this function:

6.8.2.3 int getBackCommandLine (char ∗ *command,* char ∗ *what2GetBack,* unsigned int *what2GetBackMaxSize*)

6.8.2.4 int init_dynamic_content ( )

Here is the call graph for this function:

6.8.2.5 void joystickExecute (float *x,* float *y*)

Here is the call graph for this function:

6.8.2.6 int main (int *argc,* char ∗ *argv[ ]*)

Here is the call graph for this function:

6.8.2.7 void∗ prepare_base_image (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.8.2.8 void∗ prepare_form_content_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.8.2.9 void∗ prepare_index_content_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

6.8.2.10 void∗ prepare_stats_content_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.8.2.11 void∗ prepare_top_image (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.8.2.12 void replaceChar (char ∗ *input,* char *findChar,* char *replaceWith*)

6.8.2.13 void∗ store_new_configuration_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.8.2.14 void termination_handler (int *signum*)

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

Here is the call graph for this function:

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 base_image ={0}

6.8.3.4 struct AmmServer_RH_Context chatbox ={0}

6.8.3.5 struct AmmServer_Instance∗ default_server =0

Dynamic content code ..! START!

6.8.3.6 struct AmmServer_RH_Context form ={0}

6.8.3.7 struct AmmServer_RequestOverride_Context GET_override ={{0}}

6.8.3.8 struct AmmServer_RH_Context indexPage ={0}

6.8.3.9 char∗ page =0

6.8.3.10 unsigned int pageLength =0

6.8.3.11 struct AmmServer_RH_Context random_chars ={0}

6.8.3.12 struct AmmServer_RH_Context settings ={0}

6.8.3.13 struct AmmServer_RH_Context stats ={0}

6.8.3.14 char templates_root[MAX_FILE_PATH] ="public_html/templates/"

6.8.3.15 struct AmmServer_RH_Context top_image ={0}

6.8.3.16 char webserver_root[MAX_FILE_PATH] ="public_html/"

6.9 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 ENABLE_GET_DEBUGGING 1
- #define ENABLE_STOP_PAGE 0
- #define logEcho() fprintf(stderr," Reached %s , %u \n ", __FILE__, __LINE__);
- #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 WEBSERVERROOT "public_html/"
- #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 ∗ debug_get_callback (struct AmmServer_DynamicRequest ∗rqst)
- void ∗ stop_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] =WEBSERVERROOT
- 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 stop ={0}
- struct AmmServer_RH_Context getdbg ={0}
- struct AmmServer_RH_Context stats ={0}
- struct AmmServer_RH_Context form ={0}
- struct AmmServer_RH_Context chatbox ={0}
- struct AmmServer_RH_Context fresh ={0}
- struct AmmServer_RH_Context gps ={0}
- struct AmmServer_RH_Context random_chars ={0}
- struct AmmServer_RH_Context executeScriptRC ={0}

6.9.1 Macro Definition Documentation

6.9.1.1 #define ADMIN_BINDING_PORT 8082

6.9.1.2 #define DEFAULT_BINDING_PORT 8080

6.9.1.3 #define ENABLE_ADMIN_PAGE 0

6.9.1.4 #define ENABLE_CHAT_BOX 0

6.9.1.5 #define ENABLE_GET_DEBUGGING 1

6.9.1.6 #define ENABLE_PASSWORD_PROTECTION 0

6.9.1.7 #define ENABLE_STOP_PAGE 0

6.9.1.8 #define logEcho( ) fprintf(stderr," Reached %s , %u \n ", __FILE__, __LINE__);

6.9.1.9 #define MAX_BINDING_PORT 65534

6.9.1.10 #define MAX_SCRIPT_RESPONSE_SIZE 40960

6.9.1.11 #define WEBSERVERROOT "public_html/"

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∗ debug_get_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

6.9.2.3 void∗ executeScriptFunction (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.9.2.4 void init_dynamic_content ( )

Here is the call graph for this function:

6.9.2.5 int main (int *argc,* char ∗ *argv[]*)

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

Here is the call graph for this function:

6.9.2.6 void∗ prepare_chatbox_content_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.9.2.7 void∗ prepare_form_content_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.9.2.8 void∗ prepare_gps_content_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.9.2.9 void∗ prepare_random_content_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

6.9.2.10 void∗ prepare_stats_content_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

6.9.2.11 void∗ request_override_callback (char ∗ *content*)

6.9.2.12 void∗ stop_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.9.3 Variable Documentation

6.9.3.1 char admin_root[MAX_FILE_PATH] ="admin_html/"

6.9.3.2 struct AmmServer_Instance∗ admin_server =0

6.9.3.3 struct AmmServer_RH_Context chatbox ={0}

6.9.3.4 struct AmmServer_Instance∗ default_server =0

Dynamic content code ..! START!

6.9.3.5 char∗ executeScript =0

6.9.3.6 struct AmmServer_RH_Context executeScriptRC ={0}

6.9.3.7 struct AmmServer_RH_Context form ={0}

6.9.3.8 struct AmmServer_RH_Context fresh ={0}

6.9.3.9 struct AmmServer_RequestOverride_Context GET_override ={{0}}

6.9.3.10 struct AmmServer_RH_Context getdbg ={0}

6.9.3.11 struct AmmServer_RH_Context gps ={0}

6.9.3.12 struct AmmServer_RH_Context random_chars ={0}

6.9.3.13 struct AmmServer_RH_Context stats ={0}

6.9.3.14 struct AmmServer_RH_Context stop ={0}

6.9.3.15 char templates_root[MAX_FILE_PATH] ="public_html/templates/"

6.9.3.16 char webserver_root[MAX_FILE_PATH] =WEBSERVERROOT

6.10 src/Services/CinemaPilot/main.c File Reference

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

Include dependency graph for main.c:

Data Structures

- struct playlistItem
- struct playlist

Macros

- #define DEFAULT_BINDING_PORT 8080

Enumerations

- enum stateType { STATE_UNINITIALIZED =0, STATE_PLAYING, STATE_FINISHED, NUMBER_OF_STATES }
- enum commandType {
  CMD_TYPE_NONE =0, CMD_TYPE_TRAILER, CMD_TYPE_MOVIE, CMD_TYPE_LIGHTS_ON,
  CMD_TYPE_LIGHTS_OFF, CMD_TYPE_SOUND_ON, CMD_TYPE_SOUND_OFF, CMD_TYPE_INTERMISSION,
  CMD_TYPE_BELL_ON, CMD_TYPE_BELL_OFF, NUMBER_OF_COMMANDS }

Functions

- int issueCommandToMplayer (const char ∗pathToPipe, const char ∗command)
- int pauseMplayer (const char ∗pathToPipe)
- int resumeMplayer (const char ∗pathToPipe)
- int stopMplayer (const char ∗pathToPipe)
- int intermission (unsigned int seconds)
- int startMplayer (char ∗movie, char ∗subtitles, unsigned int startAt, unsigned int duration)
- int processCommand (struct playlist ∗newMovie, struct InputParserC ∗ipc, char ∗line, unsigned int words_count)
- struct playlist ∗ readPlaylist (char ∗filename)
- int executePlaylistCurrentItem (struct playlist ∗thePlaylist)
- int executePlaylist (struct playlist ∗thePlaylist)
- int keepalivePlaylist (struct playlist ∗thePlaylist)
- void ∗ prepare_stats_content_callback (struct AmmServer_DynamicRequest ∗rqst)
- void ∗ prepare_remoteControl_callback (struct AmmServer_DynamicRequest ∗rqst)
- void ∗ prepare_indexPage (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/cinemaPilot/"
- char templates_root [MAX_FILE_PATH] ="public_html/templates/"
- char mplayerControllerPath [MAX_FILE_PATH] ="/home/ammar/Videos/videoController"
- char fullScreenViewerPath [MAX_FILE_PATH] ="/home/ammar/Documents/Programming/AmmarServer/src/Services/CinemaPilot"
- struct AmmServer_Instance ∗ default_server =0
- struct AmmServer_RequestOverride_Context GET_override ={{0}}
- struct AmmServer_RH_Context indexPage ={0}
- struct AmmServer_RH_Context remoteControl ={0}
- struct AmmServer_RH_Context random_chars ={0}
- struct AmmServer_RH_Context stats ={0}
- struct playlist ∗ movieList ={0}

6.10.1 Macro Definition Documentation

6.10.1.1 #define DEFAULT_BINDING_PORT 8080

6.10.2 Enumeration Type Documentation

6.10.2.1 enum commandType

Enumerator

CMD_TYPE_NONE

CMD_TYPE_TRAILER

CMD_TYPE_MOVIE

CMD_TYPE_LIGHTS_ON

CMD_TYPE_LIGHTS_OFF

CMD_TYPE_SOUND_ON

CMD_TYPE_SOUND_OFF

CMD_TYPE_INTERMISSION

CMD_TYPE_BELL_ON

CMD_TYPE_BELL_OFF

NUMBER_OF_COMMANDS

6.10.2.2 enum stateType

Enumerator

STATE_UNINITIALIZED

STATE_PLAYING

STATE_FINISHED

NUMBER_OF_STATES

6.10.3 Function Documentation

6.10.3.1 void close_dynamic_content ( )

Here is the call graph for this function:

6.10.3.2 int executePlaylist (struct playlist ∗ *thePlaylist*)

Here is the call graph for this function:

6.10.3.3 int executePlaylistCurrentItem (struct playlist ∗ *thePlaylist*)

Here is the call graph for this function:

6.10.3.4 void init_dynamic_content ( )

Here is the call graph for this function:

6.10.3.5 int intermission (unsigned int *seconds*)

6.10.3.6 int issueCommandToMplayer (const char ∗ *pathToPipe,* const char ∗ *command*)

6.10.3.7 int keepalivePlaylist (struct playlist ∗ *thePlaylist*)

6.10.3.8 int main (int *argc,* char ∗ *argv[ ]*)

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

Here is the call graph for this function:

6.10.3.9 int pauseMplayer (const char ∗ *pathToPipe*)

Here is the call graph for this function:

6.10.3.10 void∗ prepare_indexPage (struct AmmServer_DynamicRequest ∗ *rqst*)

6.10.3.11 void∗ prepare_random_content_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

6.10.3.12 void∗ prepare_remoteControl_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.10.3.13 void∗ prepare_stats_content_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

6.10.3.14 int processCommand (struct playlist ∗ *newMovie,* struct InputParserC ∗ *ipc,* char ∗ *line,* unsigned int *words_count*)

Here is the call graph for this function:

6.10.3.15 struct playlist∗ readPlaylist (char ∗ *filename*)

Here is the call graph for this function:

6.10.3.16 void request_override_callback (void ∗ *request*)

6.10.3.17 int resumeMplayer (const char ∗ *pathToPipe*)

Here is the call graph for this function:

6.10.3.18 int startMplayer (char ∗ *movie,* char ∗ *subtitles,* unsigned int *startAt,* unsigned int *duration*)

Here is the call graph for this function:

6.10.3.19 int stopMplayer (const char ∗ *pathToPipe*)

Here is the call graph for this function:

6.10.4 Variable Documentation

6.10.4.1 struct AmmServer_Instance∗ default_server =0

Dynamic content code ..! START!

6.10.4.2 char fullScreenViewerPath[MAX_FILE_PATH] ="/home/ammar/Documents/Programming/AmmarServer/src/Services/CinemaPilot"

6.10.4.3 struct AmmServer_RequestOverride_Context GET_override ={{0}}

6.10.4.4 struct AmmServer_RH_Context indexPage ={0}

6.10.4.5 struct playlist∗ movieList ={0}

6.10.4.6 char mplayerControllerPath[MAX_FILE_PATH] ="/home/ammar/Videos/videoController"

6.10.4.7 struct AmmServer_RH_Context random_chars ={0}

6.10.4.8 struct AmmServer_RH_Context remoteControl ={0}

6.10.4.9 struct AmmServer_RH_Context stats ={0}

6.10.4.10 char templates_root[MAX_FILE_PATH] ="public_html/templates/"

6.10.4.11 char webserver_root[MAX_FILE_PATH] ="public_html/cinemaPilot/"

6.11 src/Services/GeoPosShare/main.c File Reference

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

Include dependency graph for main.c:

Macros

- #define MAX_BINDING_PORT 65534
- #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_BINDING_PORT 8081
- #define ADMIN_BINDING_PORT 8082
- #define ENABLE_ADMIN_PAGE 0

Functions

- int appendGPS_OSM_Format (char ∗filename, char ∗from, char ∗message, char ∗latitude, char ∗longitude)
- int appendGPSMessage (char ∗filename, char ∗from, char ∗message, char ∗latitude, char ∗longitude)
- void ∗ prepare_gps_content_callback (struct AmmServer_DynamicRequest ∗rqst)
- void ∗ request_override_callback (char ∗content)
- void ∗ prepare_apk_link (struct AmmServer_DynamicRequest ∗rqst)
- void ∗ prepare_indexPage (struct AmmServer_DynamicRequest ∗rqst)
- void ∗ prepare_interestPoints (struct AmmServer_DynamicRequest ∗rqst)
- 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/geoPosShare/"
- 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 interestPoints ={0}
- struct AmmServer_RH_Context indexPage ={0}
- struct AmmServer_RH_Context android ={0}
- struct AmmServer_RH_Context apk ={0}
- struct AmmServer_RH_Context gps ={0}

6.11.1 Macro Definition Documentation

6.11.1.1 #define ADMIN_BINDING_PORT 8082

6.11.1.2 #define DEFAULT_BINDING_PORT 8081

6.11.1.3 #define ENABLE_ADMIN_PAGE 0

6.11.1.4 #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.

6.11.1.5 #define MAX_BINDING_PORT 65534

6.11.2 Function Documentation

6.11.2.1 int appendGPS_OSM_Format (char ∗ *filename,* char ∗ *from,* char ∗ *message,* char ∗ *latitude,* char ∗ *longitude*)

Here is the call graph for this function:

6.11.2.2 int appendGPSMessage (char ∗ *filename,* char ∗ *from,* char ∗ *message,* char ∗ *latitude,* char ∗ *longitude*)

6.11.2.3 void close_dynamic_content ( )

Here is the call graph for this function:

6.11.2.4 void init_dynamic_content ( )

Here is the call graph for this function:

6.11.2.5 int main (int *argc,* char ∗ *argv[ ]*)

Dynamic content code ..! END ---------------------—
Here is the call graph for this function:

6.11.2.6 void∗ prepare_apk_link (struct AmmServer_DynamicRequest ∗ *rqst*)

6.11.2.7 void∗ prepare_gps_content_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.11.2.8 void∗ prepare_indexPage (struct AmmServer_DynamicRequest ∗ *rqst*)

6.11.2.9 void∗ prepare_interestPoints (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.11.2.10 void∗ request_override_callback (char ∗ *content*)

6.11.3 Variable Documentation

6.11.3.1 char admin_root[MAX_FILE_PATH] ="admin_html/"

6.11.3.2 struct AmmServer_RH_Context android ={0}

6.11.3.3 struct AmmServer_RH_Context apk ={0}

6.11.3.4 struct AmmServer_Instance∗ default_server =0

Dynamic content code ..! START!

6.11.3.5 struct AmmServer_RequestOverride_Context GET_override ={{0}}

6.11.3.6 struct AmmServer_RH_Context gps ={0}

6.11.3.7 struct AmmServer_RH_Context indexPage ={0}

6.11.3.8 struct AmmServer_RH_Context interestPoints ={0}

6.11.3.9 char templates_root[MAX_FILE_PATH] ="public_html/templates/"

6.11.3.10 char webserver_root[MAX_FILE_PATH] ="public_html/geoPosShare/"

6.12 src/Services/HabChan/main.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <unistd.h>
#include "../../AmmServerlib/AmmServerlib.h"
#include "state.h"
#include "thread.h"
#include "board.h"
#include "postReceiver.h"
```

Include dependency graph for main.c:

Macros

- #define MAX_BINDING_PORT 65534
- #define DEFAULT_BINDING_PORT 8080
- #define ADMIN_BINDING_PORT 8080
- #define WEBSERVERROOT "data/"
- #define MAX_SCRIPT_RESPONSE_SIZE 40960

Functions

- void init_dynamic_content ()
- void close_dynamic_content ()
- int main (int argc, char ∗argv[ ])

Variables

- char webserver_root [MAX_FILE_PATH] =WEBSERVERROOT
- char templates_root [MAX_FILE_PATH] =WEBSERVERROOT "/templates/"
- struct AmmServer_RH_Context boardIndexView ={0}
- struct AmmServer_RH_Context threadIndexView ={0}
- struct AmmServer_RH_Context threadView ={0}
- struct AmmServer_RH_Context postReceiver ={0}

6.12.1 Macro Definition Documentation

6.12.1.1 #define ADMIN_BINDING_PORT 8080

6.12.1.2 #define DEFAULT_BINDING_PORT 8080

6.12.1.3 #define MAX_BINDING_PORT 65534

6.12.1.4 #define MAX_SCRIPT_RESPONSE_SIZE 40960

6.12.1.5 #define WEBSERVERROOT "data/"

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[ ]*)

Here is the call graph for this function:

6.12.3 Variable Documentation

6.12.3.1 struct AmmServer_RH_Context boardIndexView ={0}

6.12.3.2 struct AmmServer_RH_Context postReceiver ={0}

6.12.3.3 char templates_root[MAX_FILE_PATH] =WEBSERVERROOT "/templates/"

6.12.3.4 struct AmmServer_RH_Context threadIndexView ={0}

6.12.3.5 struct AmmServer_RH_Context threadView ={0}

6.12.3.6 char webserver_root[MAX_FILE_PATH] =WEBSERVERROOT

6.13 src/Services/MyBlog/main.c File Reference

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

Include dependency graph for main.c:

Macros

- #define TEST_INDEX_GENERATION_ONLY 0
- #define DEFAULT_BINDING_PORT 8080

Functions

- 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] ="src/Services/MyBlog/res/"
- 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}

6.13.1 Macro Definition Documentation

6.13.1.1 #define DEFAULT_BINDING_PORT 8080

6.13.1.2 #define TEST_INDEX_GENERATION_ONLY 0

6.13.2 Function Documentation

6.13.2.1 void close_dynamic_content ( )

Here is the call graph for this function:

6.13.2.2 void init_dynamic_content ( )

Here is the call graph for this function:

6.13.2.3 int main (int *argc,* char ∗ *argv[ ]*)

Here is the call graph for this function:

6.13.2.4 void∗ prepare_random_content_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

6.13.2.5 void request_override_callback (void ∗ *request*)

6.13.3 Variable Documentation

6.13.3.1 struct AmmServer_Instance∗ default_server =0

Dynamic content code ..! START!

6.13.3.2 struct AmmServer_RequestOverride_Context GET_override ={{0}}

6.13.3.3 struct AmmServer_RH_Context random_chars ={0}

6.13.3.4 struct AmmServer_RH_Context stats ={0}

6.13.3.5 char templates_root[MAX_FILE_PATH] ="public_html/templates/"

6.13.3.6 char webserver_root[MAX_FILE_PATH] ="src/Services/MyBlog/res/"

6.14 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}

6.14.1 Macro Definition Documentation

6.14.1.1 #define DEFAULT_BINDING_PORT 8081

6.14.2 Function Documentation

6.14.2.1 void close_dynamic_content ( )

Here is the call graph for this function:

6.14.2.2 void init_dynamic_content ( )

Here is the call graph for this function:

6.14.2.3 int main (int *argc,* char ∗ *argv[ ]*)

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

Here is the call graph for this function:

6.14.2.4 void∗ prepare_stats_content_callback (struct **AmmServer_DynamicRequest** ∗ *rqst*)

6.14.2.5 void∗ processUploadCallback (struct **AmmServer_DynamicRequest** ∗ *rqst*)

Here is the call graph for this function:

6.14.2.6 void request_override_callback (void ∗ *request*)

Here is the call graph for this function:

6.14.3 Variable Documentation

6.14.3.1 struct **AmmServer_Instance**∗ default_server =0

Dynamic content code ..! START!

6.14.3.2 struct **AmmServer_RequestOverride_Context** GET_override ={{0}}

6.14.3.3 struct **AmmServer_RH_Context** stats ={0}

6.14.3.4 char templates_root[**MAX_FILE_PATH**] ="public_html/templates/"

6.14.3.5 struct **AmmServer_RH_Context** uploadProcessor ={0}

6.14.3.6 char webserver_root[**MAX_FILE_PATH**] ="src/MyLoader/htmlData/"

6.15 src/Services/MyRemoteDesktop/main.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <unistd.h>
#include "../../AmmServerlib/AmmServerlib.h"
#include "xwd-1.0.5/XwdLib.h"
#include "../../AmmCaptcha/AmmCaptcha.h"
```

Include dependency graph for main.c:

Macros

- #define XWDLIB_BRIDGE 1
- #define ALLOW_REMOTE_CONTROL 1
- #define DEFAULT_BINDING_PORT 8080

Functions

- void ∗ prepare_screen_content_callback (struct AmmServer_DynamicRequest ∗rqst)
- void ∗ prepare_index_content_callback (struct AmmServer_DynamicRequest ∗rqst)
- void ∗ prepare_command_content_callback (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] ="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 indexPageContext ={0}
- struct AmmServer_RH_Context screenContext ={0}
- struct AmmServer_RH_Context commandContext ={0}
- char indexPagePath [128] ="src/Services/MyRemoteDesktop/res/remotedesktop.html"
- char ∗ indexPage =0
- unsigned int indexPageLength =0

6.15.1 Macro Definition Documentation

6.15.1.1 #define ALLOW_REMOTE_CONTROL 1

6.15.1.2 #define DEFAULT_BINDING_PORT 8080

6.15.1.3 #define XWDLIB_BRIDGE 1

6.15.2 Function Documentation

6.15.2.1 void close_dynamic_content ( )

Here is the call graph for this function:

6.15.2.2 void init_dynamic_content ( )

Here is the call graph for this function:

6.15.2.3 int main (int *argc*, char ∗ *argv[ ]*)

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

Here is the call graph for this function:

6.15.2.4 void∗ prepare_command_content_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.15.2.5 void∗ prepare_index_content_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

6.15.2.6 void∗ prepare_screen_content_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.15.3 Variable Documentation

6.15.3.1 struct AmmServer_RH_Context commandContext ={0}

6.15.3.2 struct AmmServer_Instance∗ default_server =0

Dynamic content code ..! START!

6.15.3.3 struct AmmServer_RequestOverride_Context GET_override ={{0}}

6.15.3.4 char∗ indexPage =0

6.15.3.5 struct AmmServer_RH_Context indexPageContext ={0}

6.15.3.6 unsigned int indexPageLength =0

6.15.3.7 char indexPagePath[128] ="src/Services/MyRemoteDesktop/res/remotedesktop.html"

6.15.3.8 struct AmmServer_RH_Context screenContext ={0}

6.15.3.9 char templates_root[MAX_FILE_PATH] ="public_html/templates/"

6.15.3.10 char webserver_root[MAX_FILE_PATH] ="public_html/"

6.16 src/Services/MyRemoteDesktop/xwd-1.0.5/main.c File Reference

```
#include <stdio.h>
#include <errno.h>
#include <X11/Xos.h>
#include <stdlib.h>
#include <X11/Xlib.h>
#include <X11/Xutil.h>
#include "X11/XWDFile.h"
#include "dsimple.h"
#include "list.h"
#include "wsutils.h"
#include "multiVis.h"
```

Include dependency graph for main.c:

Macros

- #define FEEP_VOLUME 0
- #define lowbit(x) ((x) & (∼(x) + 1))

Typedefs

- typedef unsigned long Pixel

Functions

- int main (int, char ∗∗)
- void Window_Dump (Window window, FILE ∗out, unsigned char ∗data, unsigned int ∗dataWidth, unsigned int ∗dataHeight)
- int Image_Size (XImage ∗)
- int Get_XColors (XWindowAttributes ∗, XColor ∗∗)
- void _swapshort (register char ∗, register unsigned)
- void _swaplong (register char ∗, register unsigned)
- int initXwdLib (int argc, char ∗∗argv)
- int closeXwdLib ()
- int getScreen (unsigned char ∗frame, unsigned int ∗frameWidth, unsigned int ∗frameHeight)
- void usage (void)

Variables

- int i
- Window target_win
- FILE ∗ out_file = 0
- Bool frame_only = False

6.16.1 Macro Definition Documentation

6.16.1.1 #define FEEP_VOLUME 0

6.16.1.2 #define lowbit(*x*) ((x) & (∼(x) + 1))

6.16.2 Typedef Documentation

6.16.2.1 typedef unsigned long Pixel

6.16.3 Function Documentation

6.16.3.1 void _swaplong (register char ∗ *bp,* register unsigned *n*)

6.16.3.2 void _swapshort (register char ∗ *bp,* register unsigned *n*)

6.16.3.3 int closeXwdLib ( )

6.16.3.4 int Get_XColors (XWindowAttributes ∗ *win_info,* XColor ∗∗ *colors*)

6.16.3.5 int getScreen (unsigned char ∗ *frame,* unsigned int ∗ *frameWidth,* unsigned int ∗ *frameHeight*)

Here is the call graph for this function:

6.16.3.6 int Image_Size (XImage ∗ *image*)

6.16.3.7 int initXwdLib (int *argc,* char ∗∗ *argv*)

Here is the call graph for this function:

6.16.3.8 int main (int *argc,* char ∗∗ *argv*)

Here is the call graph for this function:

6.16.3.9 void usage (void)

6.16.3.10 void Window_Dump (Window *window,* FILE ∗ *out,* unsigned char ∗ *data,* unsigned int ∗ *dataWidth,* unsigned int ∗ *dataHeight*)

Here is the call graph for this function:

6.16.4 Variable Documentation

6.16.4.1 Bool frame_only = False

6.16.4.2 int i

6.16.4.3 FILE∗ out_file = 0

6.16.4.4 Window target_win

6.17 src/Services/MyTube/main.c File Reference

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

Include dependency graph for main.c:

Macros

- #define DEFAULT_BINDING_PORT 8080
- #define DO_DYNAMIC_THUMBNAILS 1
- #define UPDATE_ALL_THUMBNAILS_ON_LAUNCH 0
- #define VIDEO_FILES_PATH_1 "/media/db46941e-4297-41d0-aa7e-659452e16780/home/guarddog/-Internet/"
- #define VIDEO_FILES_PATH_2 "/home/ammar/Videos/Internet/"
- #define VIDEO_FILES_PATH_3 "∼/Videos/"

Functions

- void ∗ serve_videofile (struct AmmServer_DynamicRequest ∗rqst)
- void ∗ serve_videopage (struct AmmServer_DynamicRequest ∗rqst)
- void ∗ serve_random_videopage (struct AmmServer_DynamicRequest ∗rqst)
- void ∗ serve_index (struct AmmServer_DynamicRequest ∗rqst)
- void ∗ serve_favicon (struct AmmServer_DynamicRequest ∗rqst)
- void ∗ serve_thumbnail (struct AmmServer_DynamicRequest ∗rqst)
- void ∗ serve_interact (struct AmmServer_DynamicRequest ∗rqst)
- int thumbnailAllVideoDatabase (struct videoCollection ∗db)
- 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 video_root [MAX_FILE_PATH] ="∼/Videos/"
- char database_root [MAX_FILE_PATH] ="∼/Videos/db/"
- struct videoCollection ∗ myTube =0
- struct AmmServer_Instance ∗ default_server =0
- struct
  AmmServer_RequestOverride_Context GET_override ={{0}}
- struct AmmServer_RH_Context random_chars ={0}
- struct AmmServer_RH_Context videoPageContext ={0}
- struct AmmServer_RH_Context videoFileContext ={0}
- struct AmmServer_RH_Context randomVideoFileContext ={0}
- struct AmmServer_RH_Context thumbnailContext ={0}
- struct AmmServer_RH_Context interactContext ={0}
- struct AmmServer_RH_Context indexContext ={0}
- struct AmmServer_RH_Context faviconContext ={0}
- struct AmmServer_MemoryHandler ∗ indexPage =0
- struct AmmServer_MemoryHandler ∗ favicon =0

6.17.1 Macro Definition Documentation

6.17.1.1 #define DEFAULT_BINDING_PORT 8080

6.17.1.2 #define DO_DYNAMIC_THUMBNAILS 1

6.17.1.3 #define UPDATE_ALL_THUMBNAILS_ON_LAUNCH 0

6.17.1.4 #define VIDEO_FILES_PATH_1 "/media/db46941e-4297-41d0-aa7e-659452e16780/home/guarddog/Internet/"

6.17.1.5 #define VIDEO_FILES_PATH_2 "/home/ammar/Videos/Internet/"

6.17.1.6 #define VIDEO_FILES_PATH_3 "∼/Videos/"

6.17.2 Function Documentation

6.17.2.1 void close_dynamic_content ( )

Here is the call graph for this function:

6.17.2.2 void init_dynamic_content ( )

Here is the call graph for this function:

6.17.2.3 int main (int *argc,* char ∗ *argv[]*)

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

Here is the call graph for this function:

6.17.2.4 void∗ serve_favicon (struct AmmServer_DynamicRequest ∗ *rqst*)

6.17.2.5 void∗ serve_index (struct AmmServer_DynamicRequest ∗ *rqst*)

6.17.2.6 void∗ serve_interact (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.17.2.7 void∗ serve_random_videopage (struct AmmServer_DynamicRequest ∗ *rqst*)

6.17.2.8 void∗ serve_thumbnail (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.17.2.9 void∗ serve_videofile (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.17.2.10 void∗ serve_videopage (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.17.2.11 int thumbnailAllVideoDatabase (struct videoCollection ∗ *db*)

Here is the call graph for this function:

6.17.3 Variable Documentation

6.17.3.1 char database_root[MAX_FILE_PATH] ="∼/Videos/db/"

6.17.3.2 struct AmmServer_Instance∗ default_server =0

Dynamic content code ..! START!

6.17.3.3 struct AmmServer_MemoryHandler∗ favicon =0

6.17.3.4 struct AmmServer_RH_Context faviconContext ={0}

6.17.3.5 struct AmmServer_RequestOverride_Context GET_override ={{0}}

6.17.3.6 struct AmmServer_RH_Context indexContext ={0}

6.17.3.7 struct AmmServer_MemoryHandler∗ indexPage =0

6.17.3.8 struct AmmServer_RH_Context interactContext ={0}

6.17.3.9 struct videoCollection∗ myTube =0

6.17.3.10 struct AmmServer_RH_Context random_chars ={0}

6.17.3.11 struct AmmServer_RH_Context randomVideoFileContext ={0}

6.17.3.12 char templates_root[MAX_FILE_PATH] ="public_html/templates/"

6.17.3.13 struct AmmServer_RH_Context thumbnailContext ={0}

6.17.3.14 char video_root[MAX_FILE_PATH] ="∼/Videos/"

6.17.3.15 struct AmmServer_RH_Context videoFileContext ={0}

6.17.3.16 struct AmmServer_RH_Context videoPageContext ={0}

6.17.3.17 char webserver_root[MAX_FILE_PATH] ="public_html/"

6.18 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)

  This is a custom resolver for requests When a new message is received this gets called and depending on the resource requested we camouflage the request in a way that we want to make urls more user friendly so a request for /whatever is converted to /go?to=whatever in ourcase :) (since we have a url shortner)
- 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

6.18.1 Macro Definition Documentation

6.18.1.1 #define DEFAULT_BINDING_PORT 8080

6.18.1.2 #define DYNAMIC_PAGES_MEMORY_COMMITED 4096

6.18.1.3 #define ENABLE_CAPTCHA_SYSTEM 1

6.18.1.4 #define LINK_ALLOCATION_STEP 5000

6.18.1.5 #define MAX_BINDING_PORT 65534

6.18.1.6 #define MAX_CAPTCHA_JPG_SIZE 10 ∗ 1024

6.18.1.7 #define MAX_LINKS 200000

6.18.1.8 #define MAX_LONG_URL_SIZE 2048

6.18.1.9 #define MAX_TO_SIZE 32

6.18.1.10 #define REGROUP_AFTER_X_UNSORTED_LINKS 1000

6.18.1.11 #define USE_BINARY_SEARCH 1

6.18.2 Function Documentation

6.18.2.1 unsigned long Add_MyURL (char ∗ *longURL,* char ∗ *shortURL,* int *saveit*)

Here is the call graph for this function:

6.18.2.2 unsigned int allocateLinksIfNeeded ( )

Here is the call graph for this function:

6.18.2.3 int Append2MyURLDBFile (char ∗ *filename,* char ∗ *longURL,* char ∗ *shortURL*)

6.18.2.4 void close_dynamic_content ( )

Here is the call graph for this function:

6.18.2.5 unsigned int Find_longURL (char ∗ *shortURL,* int ∗ *found*) `[inline]`

Here is the call graph for this function:

6.18.2.6 unsigned int Find_longURLSerial (char ∗ *shortURL,* int ∗ *found*) `[inline]`

Here is the call graph for this function:

6.18.2.7 char∗ Get_longURL (char ∗ *shortURL*)

Here is the call graph for this function:

6.18.2.8 unsigned long hashURL (char ∗ *str*)

6.18.2.9 void init_dynamic_content ( )

Here is the call graph for this function:

6.18.2.10 int is_an_unsafe_str (char ∗ *input,* unsigned int *input_length*)

6.18.2.11 int isURLDBSorted ( )

6.18.2.12 int LoadMyURLDBFile (char ∗ *filename*)

Here is the call graph for this function:

6.18.2.13 int main (int *argc,* char ∗ *argv[ ]*)

Here is the call graph for this function:

6.18.2.14 int printURLDB ( )

6.18.2.15 void resolveRequest (void ∗ *request*)

This is a custom resolver for requests When a new message is received this gets called and depending on the resource requested we camouflage the request in a way that we want to make urls more user friendly so a request for /whatever is converted to /go?to=whatever in ourcase :) (since we have a url shortner)

Here is the call graph for this function:

6.18.2.16 int ResortDB (char ∗ *db_file,* struct URLDB ∗ *links,* unsigned int *loaded_links*)

Here is the call graph for this function:

6.18.2.17 int ReWriteMyURLDBFile (char ∗ *filename,* struct URLDB ∗ *links,* unsigned int *loaded_links*)

6.18.2.18 void∗ serve_captcha_page (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.18.2.19 void∗ serve_create_url_page (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.18.2.20 void∗ serve_error_url_page (struct AmmServer_DynamicRequest ∗ *rqst*)

6.18.2.21 void∗ serve_goto_url_page (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.18.2.22 int struct_cmp_urldb_items (const void ∗ *a,* const void ∗ *b*)

6.18.3 Variable Documentation

6.18.3.1 unsigned int allocated_links =0

6.18.3.2 struct AmmServer_RH_Context captcha_url ={0}

6.18.3.3 struct AmmServer_RH_Context create_url ={0}

6.18.3.4 pthread_mutex_t db_addIDLock

6.18.3.5 char db_file[128] ="myurl.db"

6.18.3.6 pthread_mutex_t db_fileLock

6.18.3.7 char∗ default_failed = (char∗)"http://myurl.ammar.gr/error.html"

6.18.3.8 struct AmmServer_RH_Context error_url ={0}

6.18.3.9 struct AmmServer_RH_Context goto_url ={0}

6.18.3.10 char∗ indexPage =0

6.18.3.11 unsigned int indexPageLength =0

6.18.3.12 char indexPagePath[128] ="src/Services/MyURL/myurl.html"

6.18.3.13 struct URLDB∗ links =0

6.18.3.14 unsigned int loaded_links =0

6.18.3.15 struct AmmServer_Instance∗ myurl_server =0

6.18.3.16 struct AmmServer_RequestOverride_Context requestResolver ={{0}}

6.18.3.17 char service_filename[5] ="/go"

6.18.3.18 char service_filename_noslash[5] ="go"

6.18.3.19 char service_root[128] ="http://myurl.ammar.gr/go"

6.18.3.20 char service_root_withoutfilename[128] ="http://myurl.ammar.gr/"

6.18.3.21 unsigned int sorted_links =0

6.18.3.22 char templates_root[MAX_FILE_PATH] ="public_html/templates/"

6.18.3.23 char webserver_root[MAX_FILE_PATH] ="public_html/"

6.19 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}

6.19.1 Macro Definition Documentation

6.19.1.1 #define DEFAULT_BINDING_PORT 8080

6.19.2 Function Documentation

6.19.2.1 void close_dynamic_content ( )

Here is the call graph for this function:

6.19.2.2 void init_dynamic_content ( )

Here is the call graph for this function:

6.19.2.3 int main (int *argc*, char ∗ *argv[]*)

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

Here is the call graph for this function:

6.19.2.4 void∗ prepare_random_content_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

6.19.2.5 void∗ prepare_stats_content_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

6.19.2.6 void request_override_callback (void ∗ *request*)

6.19.3 Variable Documentation

6.19.3.1 struct AmmServer_Instance∗ default_server =0

Dynamic content code ..! START!

6.19.3.2 struct AmmServer_RequestOverride_Context GET_override ={{0}}

6.19.3.3 struct AmmServer_RH_Context random_chars ={0}

6.19.3.4 struct AmmServer_RH_Context stats ={0}

6.19.3.5 char templates_root[MAX_FILE_PATH] ="public_html/templates/"

6.19.3.6 char webserver_root[MAX_FILE_PATH] ="public_html/"

6.20 src/Services/SQLiteServer/main.c File Reference

```
#include <sqlite3.h>
#include "sqlite.h"
#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_cars_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}
- struct SQLiteSession sqliteSession ={0}

6.20.1 Macro Definition Documentation

6.20.1.1 #define DEFAULT_BINDING_PORT 8080

6.20.2 Function Documentation

6.20.2.1 void close_dynamic_content ( )

Here is the call graph for this function:

6.20.2.2 void init_dynamic_content ( )

Here is the call graph for this function:

6.20.2.3 int main (int *argc,* char ∗ *argv[ ]*)

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

Here is the call graph for this function:

6.20.2.4 void∗ prepare_cars_content_callback (struct **AmmServer_DynamicRequest** ∗ *rqst*)

Here is the call graph for this function:

6.20.2.5 void∗ prepare_stats_content_callback (struct **AmmServer_DynamicRequest** ∗ *rqst*)

6.20.2.6 void request_override_callback (void ∗ *request*)

6.20.3 Variable Documentation

6.20.3.1 struct **AmmServer_Instance**∗ default_server =0

Dynamic content code ..! START!

6.20.3.2 struct **AmmServer_RequestOverride_Context** GET_override ={{0}}

6.20.3.3 struct **AmmServer_RH_Context** random_chars ={0}

6.20.3.4 struct **SQLiteSession** sqliteSession ={0}

6.20.3.5 struct **AmmServer_RH_Context** stats ={0}

6.20.3.6 char templates_root[**MAX_FILE_PATH**] ="public_html/templates/"

6.20.3.7 char webserver_root[**MAX_FILE_PATH**] ="public_html/"

6.21 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[ ])

6.21.1 Function Documentation

6.21.1.1 int main (int *argc,* char ∗ *argv[ ]*)

Here is the call graph for this function:

6.22 src/UserAccounts/main.c File Reference

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

Include dependency graph for main.c:

Functions

- struct UserAccountDatabase ∗ uadb_initializeUserAccountDatabase (char ∗filename)
- int uadb_closeUserAccountDatabase (struct UserAccountDatabase ∗∗uadb)
- int uadb_authenticateUser (struct UserAccountDatabase ∗uadb, struct UserAccountAuthenticationToken ∗outputToken, UserAccount_UserID userID)
- int uadb_loginUser (struct UserAccountDatabase ∗uadb, struct UserAccountAuthenticationToken ∗outputToken, char ∗username, char ∗password, UserAccount_PasswordEncoding encoding, char ∗ip, char ∗browserFingerprint)

6.22.1 Function Documentation

6.22.1.1 int uadb_authenticateUser (struct **UserAccountDatabase** ∗ *uadb,* struct **UserAccountAuthenticationToken** ∗ *outputToken,* **UserAccount_UserID** *userID*)

6.22.1.2 int uadb_closeUserAccountDatabase (struct **UserAccountDatabase** ∗∗ *uadb*)

6.22.1.3 struct **UserAccountDatabase**∗ uadb_initializeUserAccountDatabase (char ∗ *filename*)

6.22.1.4 int uadb_loginUser (struct **UserAccountDatabase** ∗ *uadb,* struct **UserAccountAuthenticationToken** ∗ *outputToken,* char ∗ *username,* char ∗ *password,* **UserAccount_PasswordEncoding** *encoding,* char ∗ *ip,* char ∗ *browserFingerprint*)

6.23 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)

6.23.1 Macro Definition Documentation

6.23.1.1 #define DISPLAY_DEBUG_INFO 0

6.23.1.2 #define PPMREADBUFLEN 256

6.23.1.3 #define READ_CREATES_A_NEW_PIXEL_BUFFER 1

6.23.2 Function Documentation

6.23.2.1 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*)

6.23.2.2 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*)

6.23.2.3 struct **Image**∗ copyImage (struct **Image** ∗ *source*)

6.23.2.4 struct **Image**∗ createImage (unsigned int *width,* unsigned int *height,* unsigned int *depth*)

6.23.2.5 int destroyImage (struct **Image** ∗ *source*)

6.23.2.6 int ReadPPM (struct **Image** ∗ *pic,* char ∗ *filename,* char *read_only_header*)

6.23.2.7 int WritePPM (struct **Image** ∗ *pic,* char ∗ *filename*)

6.24 src/AmmCaptcha/imaging.h File Reference

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)

6.24.1 Function Documentation

6.24.1.1 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*)

6.24.1.2 struct Image∗ copyImage (struct Image ∗ *source*)

6.24.1.3 struct Image∗ createImage (unsigned int *width,* unsigned int *height,* unsigned int *depth*)

6.24.1.4 int destroyImage (struct Image ∗ *source*)

6.24.1.5 int ReadPPM (struct Image ∗ *pic,* char ∗ *filename,* char *read_only_header*)

6.24.1.6 int WritePPM (struct Image ∗ *pic,* char ∗ *filename*)

6.25 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)

6.25.1 Macro Definition Documentation

6.25.1.1 #define ABS(*num1*) ((num1) >=0 ? (num1) : (-1∗num1))

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

6.25.2 Function Documentation

6.25.2.1 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:

6.25.2.2 int warpImage (struct Image ∗ *target,* unsigned int *posX,* unsigned int *posY,* signed int *warpDeltaX,* signed int *warpDeltaY*)

Here is the call graph for this function:

6.26 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)

6.26.1 Function Documentation

6.26.1.1 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

Here is the call graph for this function:

6.26.1.2 int warpImage (struct Image ∗ *target,* unsigned int *posX,* unsigned int *posY,* signed int *warpDeltaX,* signed int *warpDeltaY*)

Here is the call graph for this function:

6.27 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 ()

6.27.1 Function Documentation

6.27.1.1 int empty_buffer (struct jpeg_compress_struct ∗ *cinfo*)

6.27.1.2 int fastJPGHeaderCheck (FILE ∗ *file*)

6.27.1.3 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

6.27.1.4 int jpegtest ( )

Here is the call graph for this function:

6.27.1.5 int ReadJPEG (char ∗ *filename,* struct Image ∗ *pic,* char *read_only_header*)

Here is the call graph for this function:

6.27.1.6 void term_buffer (struct jpeg_compress_struct ∗ *cinfo*)

6.27.1.7 int WriteJPEGFile (struct Image ∗ *pic,* char ∗ *filename*)

Here is the call graph for this function:

6.27.1.8 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

Here is the call graph for this function:

6.27.1.9 int WriteJPEGMemory (struct Image ∗ *pic,* char ∗ *mem,* unsigned long ∗ *mem_size*)

Here is the call graph for this function:

6.28 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)

6.28.1 Macro Definition Documentation

6.28.1.1 #define USE_JPG_FILES 1

6.28.2 Function Documentation

6.28.2.1 int ReadJPEG (char ∗ *filename,* struct Image ∗ *pic,* char *read_only_header*)

Here is the call graph for this function:

6.28.2.2 int WriteJPEGFile (struct Image ∗ *pic,* char ∗ *filename*)

Here is the call graph for this function:

6.28.2.3 int WriteJPEGMemory (struct Image ∗ *pic,* char ∗ *mem,* unsigned long ∗ *mem_size*)

Here is the call graph for this function:

6.29 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_MemoryHandler

  A Wrapper around a memory buffer that enables house keeping for reallocations etc.
- 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 133

  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 }

  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.*
- 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_RANDOMIZE_ETAG_BEGINNING, AMMSET_TEST }

  Enumerator for calls AmmServer_GetIntSettingValue and AmmServer_SetIntSettingValue.
- enum AmmServStrSettings { AMMSET_USERNAME_STR =0, AMMSET_PASSWORD_STR, AMMSET_TESTSTR }

  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 (const char ∗name, const char ∗ip, unsigned int port, const char ∗conf_file, const char ∗web_root_path, const char ∗templates_root_path)

  Start a Web Server , allocate memory , bind ports and return its instance..
- struct AmmServer_Instance ∗ AmmServer_StartWithArgs (const char ∗name, int argc, char ∗∗argv, const char ∗ip, unsigned int port, const char ∗conf_file, const char ∗web_root_path, const 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_DynamicRequestReturnFile (struct AmmServer_DynamicRequest ∗rqst, const char ∗filename)

  Return a file instead of a Dynamic Request.
- int AmmServer_AddRequestHandler (struct AmmServer_Instance ∗instance, struct AmmServer_RequestOverride_Context ∗RequestOverrideContext, const 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, const char ∗resource_name, const 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, const 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, const 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, const 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, const 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, const 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, const char ∗var_id_IN, 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, const 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 (const 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, const 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 (const 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_ExecuteCommandLineNum (const char ∗command, char ∗what2GetBack, unsigned int what2GetBackMaxSize, unsigned int lineNumber)

  Execute a command and copy its output line to the provided buffer.
- int AmmServer_ExecuteCommandLine (const char ∗command, char ∗what2GetBack, unsigned int what2GetBackMaxSize)

  Execute a command and copy its output to the provided buffer.
- void AmmServer_ReplaceCharInString (char ∗input, char findChar, char replaceWith)

  Hot-Replace a character inside a memory block , typically used to replace characters like '+' with ' '.
- int AmmServer_ReplaceVarInMemoryFile (char ∗page, unsigned int pageLength, const char ∗var, const 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.
- int AmmServer_ReplaceAllVarsInMemoryFile (char ∗page, unsigned int instances, unsigned int pageLength, const char ∗var, const char ∗value)

  Hot-Replace ALL variables inside a memory block , typically used to replace placeholders inside text files , like $$$$$$$$NAME$$$$$$$$, the value should be smaller or equal to the var being replaced.
- char ∗ AmmServer_ReadFileToMemory (const char ∗filename, unsigned int ∗length)

  Read a file and store it to a freshly allocated memory block.
- int AmmServer_WriteFileFromMemory (const char ∗filename, char ∗memory, unsigned int memoryLength)

  Dump a memory block to a file.
- struct AmmServer_MemoryHandler ∗ AmmServer_ReadFileToMemoryHandler (const char ∗filename)

  Read a file and store it to a freshly allocated memory handler context.
- struct AmmServer_MemoryHandler ∗ AmmServer_CopyMemoryHandler (struct AmmServer_MemoryHandler ∗inpt)

  Copy a memory handler.
- int AmmServer_CopyOverlappingDataContent (unsigned char ∗buffer, unsigned int totalSize, unsigned char ∗from, unsigned char ∗to, unsigned int blockSize)

  Copy Content from one place of a buffer to another using an intermediate buffer..
- int AmmServer_InjectDataToBuffer (unsigned char ∗entryPoint, unsigned char ∗data, struct AmmServer_MemoryHandler ∗mh)

  Search for entryPoint pattern in buffer , and inject data there..!
- int AmmServer_ReplaceVarInMemoryHandler (struct AmmServer_MemoryHandler ∗mh, const char ∗var, const char ∗value)
- int AmmServer_ReplaceAllVarsInMemoryHandler (struct AmmServer_MemoryHandler ∗mh, unsigned int instances, const char ∗var, const char ∗value)
- struct AmmServer_MemoryHandler ∗ AmmServer_AllocateMemoryHandler (unsigned int initialBufferLength, unsigned int growStep)
- int AmmServer_FreeMemoryHandler (struct AmmServer_MemoryHandler ∗∗mh)
- 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 (const char ∗filename)

  Check if directory Exists.
- int AmmServer_FileExists (const char ∗filename)

  Check if file Exists.
- int AmmServer_FileIsVideo (const char ∗filename)

  Check if file is a video.
- int AmmServer_EraseFile (const char ∗filename)

  Erase a File.
- unsigned int AmmServer_StringIsHTMLSafe (const 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.

6.29.1 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

6.29.2 Macro Definition Documentation

6.29.2.1 #define AMMAR_SERVER_HTTP_HEADER_SPEC 133

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

6.29.2.2 #define MAX_FILE_PATH 1024

6.29.2.3 #define MAX_INSTANCE_NAME_STRING 128

6.29.2.4 #define MAX_IP_STRING_SIZE 32

6.29.2.5 #define MAX_QUERY 2048

6.29.2.6 #define MAX_RESOURCE 2048

6.29.2.7 #define POPEN_BUFFER_SIZE 256

Size for popen replies.

6.29.3 Enumeration Type Documentation

6.29.3.1 enum AmmServInfos

Enumerator for calls AmmServer_GetInfo.

Enumerator

AMMINF_ACTIVE_CLIENTS

AMMINF_ACTIVE_THREADS

6.29.3.2 enum AmmServSettings

Enumerator for calls AmmServer_GetIntSettingValue and AmmServer_SetIntSettingValue.

Enumerator

AMMSET_PASSWORD_PROTECTION

AMMSET_RANDOMIZE_ETAG_BEGINNING

AMMSET_TEST

6.29.3.3 enum AmmServStrSettings

Enumerator for calls AmmServer_GetStrSettingValue and AmmServer_SetStrSettingValue.

Enumerator

AMMSET_USERNAME_STR

AMMSET_PASSWORD_STR

AMMSET_TESTSTR

6.29.3.4 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

6.29.3.5 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

6.29.4 Function Documentation

6.29.4.1 int _FILES (struct AmmServer_Instance ∗ *instance,* struct AmmServer_DynamicRequest ∗ *rqst,* const 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:

6.29.4.2 int _GET (struct AmmServer_Instance ∗ *instance,* struct AmmServer_DynamicRequest ∗ *rqst,* const 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:

6.29.4.3 int _POST (struct AmmServer_Instance ∗ *instance,* struct AmmServer_DynamicRequest ∗ *rqst,* const 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:

6.29.4.4 int AmmServer_AddRequestHandler (struct AmmServer_Instance ∗ *instance,* struct AmmServer_RequestOverride_Context ∗ *RequestOverrideContext,* const 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
A	AmmServer_RequestOverride_Context to be populated
Request	Type
Pointer	to function callback

Return values

1=Success,0=Fail	

Here is the call graph for this function:

6.29.4.5 int AmmServer_AddResourceHandler (struct AmmServer_Instance ∗ *instance,* struct AmmServer_RH_Context ∗ *context,* const char ∗ *resource_name,* const 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:

6.29.4.6 struct **AmmServer_MemoryHandler**∗ AmmServer_AllocateMemoryHandler (unsigned int *initialBufferLength,* unsigned int *growStep*)

6.29.4.7 int AmmServer_CheckIfHeaderBinaryAreTheSame (int *headerSpec*)

Internal Check to compare against changes of the header files.

Parameters

Header	(should be AMMAR_SERVER_HTTP_HEADER_SPEC)

Return values

1=Success,0=Failure	

6.29.4.8 struct **AmmServer_MemoryHandler**∗ AmmServer_CopyMemoryHandler (struct **AmmServer_MemoryHandler** ∗ *inpt*)

Copy a memory handler.

Parameters

Input	memory handle

Return values

Pointer	to the new memory handler or 0=Failed

6.29.4.9 int AmmServer_CopyOverlappingDataContent (unsigned char ∗ *buffer,* unsigned int *totalSize,* unsigned char ∗ *from,* unsigned char ∗ *to,* unsigned int *blockSize*)

Copy Content from one place of a buffer to another using an intermediate buffer..

Parameters

Original	Buffer
Size	of Original Buffer
Pointer	to the start of the source of the copy
Pointer	to the start of the destination of the copy
Size	of data to copy

Return values

1=Ok,0=Failed	

Here is the call graph for this function:

6.29.4.10 int AmmServer_DirectoryExists (const 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:

6.29.4.11 int AmmServer_DoNOTCacheResource (struct **AmmServer_Instance** ∗ *instance,* const 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:

6.29.4.12 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:

6.29.4.13 int AmmServer_DynamicRequestReturnFile (struct **AmmServer_DynamicRequest** ∗ *rqst,* const char ∗ *filename*)

Return a file instead of a Dynamic Request.

Parameters

An	AmmarServer Request
File	to serve

Return values

1=Running,0=Stopped	

6.29.4.14 int AmmServer_EraseFile (const char ∗ *filename*)

Erase a File.

Parameters

Path	to file

Return values

1=Success,0=Failure	

6.29.4.15 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	printf (`http://www.cplusplus.com/reference/cstdio/printf/`)
Arbitrary	number of other parameters that where defined in format

Here is the call graph for this function:

6.29.4.16 int AmmServer_ExecuteCommandLine (const 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

Here is the call graph for this function:

6.29.4.17 int AmmServer_ExecuteCommandLineNum (const char ∗ *command,* char ∗ *what2GetBack,* unsigned int *what2GetBackMaxSize,* unsigned int *lineNumber*)

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

Parameters

Command	to execute
Allocated	memory to store the result
Size	of Allocated memory
Number	of line we want to get back

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

6.29.4.18 int AmmServer_FileExists (const 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:

6.29.4.19 int AmmServer_FileIsVideo (const char ∗ *filename*)

Check if file is a video.

Parameters

Path	to file

Return values

1=Exists,0=Does	not Exist

Here is the call graph for this function:

6.29.4.20 int AmmServer_FILES (struct AmmServer_Instance ∗ *instance,* struct AmmServer_DynamicRequest ∗ *rqst,* const 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	

6.29.4.21 int AmmServer_FreeMemoryHandler (struct AmmServer_MemoryHandler ∗∗ *mh*)

6.29.4.22 int AmmServer_GETArg (struct AmmServer_Instance ∗ *instance,* struct AmmServer_DynamicRequest ∗ *rqst,* const 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:

6.29.4.23 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

6.29.4.24 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

6.29.4.25 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

6.29.4.26 int AmmServer_InjectDataToBuffer (unsigned char ∗ *entryPoint,* unsigned char ∗ *data,* struct AmmServer_MemoryHandler ∗ *mh*)

Search for entryPoint pattern in buffer , and inject data there..!

Parameters

String	to find in buffer and replace with new content
Data	we want to inject
Memory	Handler for Buffer we want to inject to , see struct AmmServer_MemoryHandler

Return values

1=Ok,0=Failed	

Here is the call graph for this function:

6.29.4.27 int AmmServer_POSTArg (struct AmmServer_Instance ∗ *instance,* struct AmmServer_DynamicRequest ∗ *rqst,* const 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:

6.29.4.28 char∗ AmmServer_ReadFileToMemory (const 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:

6.29.4.29 struct AmmServer_MemoryHandler∗ AmmServer_ReadFileToMemoryHandler (const char ∗ *filename*)

Read a file and store it to a freshly allocated memory handler context.

Parameters

Input	Filename

Return values

Pointer	to the new memory handler or 0=Failed

Here is the call graph for this function:

6.29.4.30 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:

6.29.4.31 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:

6.29.4.32 int AmmServer_ReplaceAllVarsInMemoryFile (char ∗ *page*, unsigned int *instances*, unsigned int *pageLength*, const char ∗ *var*, const char ∗ *value*)

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

Parameters

Pointer	to memory that contains the document
Maximum	number of Variable instances , 0 means infinite (until the end of the memory buffer)..
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 , this should be handled

Here is the call graph for this function:

6.29.4.33 int AmmServer_ReplaceAllVarsInMemoryHandler (struct **AmmServer_MemoryHandler** ∗ *mh*, unsigned int *instances*, const char ∗ *var*, const char ∗ *value*)

Here is the call graph for this function:

6.29.4.34 void AmmServer_ReplaceCharInString (char ∗ *input*, char *findChar*, char *replaceWith*)

Hot-Replace a character inside a memory block , typically used to replace characters like '+' with ' '.

Parameters

Pointer	to memory that contains the null terminated string
Character	to be replaced
What	to replace the character with

6.29.4.35 int AmmServer_ReplaceVarInMemoryFile (char * *page,* unsigned int *pageLength,* const char * *var,* const 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 , this should be handled

Here is the call graph for this function:

6.29.4.36 int AmmServer_ReplaceVarInMemoryHandler (struct **AmmServer_MemoryHandler** * *mh,* const char * *var,* const char * *value*)

Here is the call graph for this function:

6.29.4.37 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:

6.29.4.38 int AmmServer_SaveDynamicRequest (const 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:

6.29.4.39 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

6.29.4.40 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

Here is the call graph for this function:

6.29.4.41 int AmmServer_SetStrSettingValue (struct **AmmServer_Instance** ∗ *instance,* unsigned int *set_type,* const 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:

6.29.4.42 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

6.29.4.43 struct **AmmServer_Instance**∗ AmmServer_Start (const char ∗ *name,* const char ∗ *ip,* unsigned int *port,* const char ∗ *conf_file,* const char ∗ *web_root_path,* const 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:

6.29.4.44 struct **AmmServer_Instance**∗ AmmServer_StartAdminInstance (const 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

6.29.4.45 struct **AmmServer_Instance**∗ AmmServer_StartWithArgs (const char ∗ *name,* int *argc,* char ∗∗ *argv,* const char ∗ *ip,* unsigned int *port,* const char ∗ *conf_file,* const char ∗ *web_root_path,* const 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:

6.29.4.46 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:

6.29.4.47 unsigned int AmmServer_StringIsHTMLSafe (const 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

1=Safe,0=Unsafe	

6.29.4.48 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	printf (`http://www.cplusplus.com/reference/cstdio/printf/`)
Arbitrary	number of other parameters that where defined in format

Here is the call graph for this function:

6.29.4.49 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.

6.29.4.50 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	printf (`http://www.cplusplus.com/reference/cstdio/printf/`)
Arbitrary	number of other parameters that where defined in format

Here is the call graph for this function:

6.29.4.51 int AmmServer_WriteFileFromMemory (const 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	

Here is the call graph for this function:

6.30 src/AmmServerlib/AString/AString.c File Reference

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

Include dependency graph for AString.c:

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 ∗/

Functions

- int astringReplaceVarInMemoryFile (char ∗page, unsigned int pageLength, const char ∗var, const char ∗value)
- int astringReplaceAllInstancesOfVarInMemoryFile (char ∗page, unsigned int instances, unsigned int pageLength, const char ∗var, const char ∗value)
- char ∗ astringReadFileToMemory (const char ∗filename, unsigned int ∗length)
- int astringWriteFileFromMemory (const char ∗filename, char ∗memory, unsigned int memoryLength)
- int astringCopyOverlappingDataContent (unsigned char ∗buffer, unsigned int totalSize, unsigned char ∗from, unsigned char ∗to, unsigned int blockSize)

- int astringInjectDataToBuffer (unsigned char ∗entryPoint, unsigned char ∗data, unsigned char ∗buffer, unsigned int currentBufferLength, unsigned int totalBufferLength)
- int myStupidMemcpy (char ∗target, char ∗source, unsigned int sourceLength)
- int astringInjectDataToMemoryHandler (struct AmmServer_MemoryHandler ∗mh, const char ∗var, const char ∗value)

    Inject a String inside a memory handler.

6.30.1 Macro Definition Documentation

6.30.1.1 #define BLACK "\033[30m" /∗ Black ∗/

6.30.1.2 #define GREEN "\033[32m" /∗ Green ∗/

6.30.1.3 #define NORMAL "\033[0m"

6.30.1.4 #define RED "\033[31m" /∗ Red ∗/

6.30.1.5 #define YELLOW "\033[33m" /∗ Yellow ∗/

6.30.2 Function Documentation

6.30.2.1 int astringCopyOverlappingDataContent (unsigned char ∗ *buffer,* unsigned int *totalSize,* unsigned char ∗ *from,* unsigned char ∗ *to,* unsigned int *blockSize*)

6.30.2.2 int astringInjectDataToBuffer (unsigned char ∗ *entryPoint,* unsigned char ∗ *data,* unsigned char ∗ *buffer,* unsigned int *currentBufferLength,* unsigned int *totalBufferLength*)

Here is the call graph for this function:

6.30.2.3 int astringInjectDataToMemoryHandler (struct AmmServer_MemoryHandler ∗ *mh,* const char ∗ *var,* const char ∗ *value*)

Inject a String inside a memory handler.

Parameters

Pointer	to a MemoryHandler struct that contains the buffer we want to modify
needle	we want to search in the haystack to replace
String	that will replace existing one

Return values

1=Success,0=Fail	

Bug This does not yet reallocate the buffer to make it bigger in case it is not big enough to accomodate the new string..

6.30.2.4 char∗ astringReadFileToMemory (const char ∗ *filename,* unsigned int ∗ *length*)

6.30.2.5 int astringReplaceAllInstancesOfVarInMemoryFile (char ∗ *page,* unsigned int *instances,* unsigned int *pageLength,* const char ∗ *var,* const char ∗ *value*)

Here is the call graph for this function:

6.30.2.6 int astringReplaceVarInMemoryFile (char ∗ *page,* unsigned int *pageLength,* const char ∗ *var,* const char ∗ *value*)

6.30.2.7 int astringWriteFileFromMemory (const char ∗ *filename,* char ∗ *memory,* unsigned int *memoryLength*)

6.30.2.8 int myStupidMemcpy (char ∗ *target,* char ∗ *source,* unsigned int *sourceLength*)

6.31 src/AmmServerlib/AString/AString.h File Reference

A small toolset to handle long strings manage memory and append,inject other strings inside them.

```
#include "../AmmServerlib.h"
```

Include dependency graph for AString.h: This graph shows which files directly or indirectly include this file:

Functions

- int astringReplaceVarInMemoryFile (char ∗page, unsigned int pageLength, const char ∗var, const char ∗value)
- int astringReplaceAllInstancesOfVarInMemoryFile (char ∗page, unsigned int instances, unsigned int pageLength, const char ∗var, const char ∗value)
- char ∗ astringReadFileToMemory (const char ∗filename, unsigned int ∗length)
- int astringWriteFileFromMemory (const char ∗filename, char ∗memory, unsigned int memoryLength)
- int astringCopyOverlappingDataContent (unsigned char ∗buffer, unsigned int totalSize, unsigned char ∗from, unsigned char ∗to, unsigned int blockSize)
- int astringInjectDataToBuffer (unsigned char ∗entryPoint, unsigned char ∗data, unsigned char ∗buffer, unsigned int currentBufferLength, unsigned int totalBufferLength)
- int astringInjectDataToMemoryHandler (struct AmmServer_MemoryHandler ∗mh, const char ∗var, const char ∗value)

  Inject a String inside a memory handler.

6.31.1 Detailed Description

A small toolset to handle long strings manage memory and append,inject other strings inside them.

Author

Ammar Qammaz (AmmarkoV)

6.31.2 Function Documentation

6.31.2.1 int astringCopyOverlappingDataContent (unsigned char ∗ *buffer,* unsigned int *totalSize,* unsigned char ∗ *from,* unsigned char ∗ *to,* unsigned int *blockSize*)

6.31.2.2 int astringInjectDataToBuffer (unsigned char ∗ *entryPoint,* unsigned char ∗ *data,* unsigned char ∗ *buffer,* unsigned int *currentBufferLength,* unsigned int *totalBufferLength*)

Here is the call graph for this function:

6.31.2.3 int astringInjectDataToMemoryHandler (struct AmmServer_MemoryHandler ∗ *mh,* const char ∗ *var,* const char ∗ *value*)

Inject a String inside a memory handler.

Parameters

Pointer	to a MemoryHandler struct that contains the buffer we want to modify
needle	we want to search in the haystack to replace
String	that will replace existing one

Return values

1=Success,0=Fail	

Bug This does not yet reallocate the buffer to make it bigger in case it is not big enough to accomodate the new string..

6.31.2.4 char∗ astringReadFileToMemory (const char ∗ *filename,* unsigned int ∗ *length*)

6.31.2.5 int astringReplaceAllInstancesOfVarInMemoryFile (char ∗ *page,* unsigned int *instances,* unsigned int *pageLength,* const char ∗ *var,* const char ∗ *value*)

Here is the call graph for this function:

6.31.2.6 int astringReplaceVarInMemoryFile (char ∗ *page,* unsigned int *pageLength,* const char ∗ *var,* const char ∗ *value*)

6.31.2.7 int astringWriteFileFromMemory (const char ∗ *filename,* char ∗ *memory,* unsigned int *memoryLength*)

6.32 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_isClientAllowedToMakeAConnection (char ∗ip)
- 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.

6.32.1 Macro Definition Documentation

6.32.1.1 #define COMPILE_WITH_CLIENT_LIST 1

6.32.2 Function Documentation

6.32.2.1 int clientList_close (struct clientListContext ∗ *clientList*)

Close and destroy client list.

Parameters

ClientList	to destroy

Return values

1=Success,0=Failure	

6.32.2.2 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

6.32.2.3 struct clientListContext∗ clientList_initialize ( )

Create , allocate and return a client list.

Return values

Pointer	to a freshly allocated client list or 0=Failure

6.32.2.4 int clientList_isClientAllowedToMakeAConnection (char ∗ *ip*)

6.32.2.5 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	

6.32.2.6 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

1=Banned,0=OK	

6.32.2.7 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	

6.33 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_isClientAllowedToMakeAConnection (char ∗ip)
- 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.

6.33.1 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

6.33.2 Typedef Documentation

6.33.2.1 typedef unsigned int **clientID**

Typedef to make clientID stand out.

6.33.3 Function Documentation

6.33.3.1 int clientList_close (struct **clientListContext** ∗ *clientList*)

Close and destroy client list.

Parameters

ClientList	to destroy

Return values

1=Success,0=Failure	

6.33.3.2 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

6.33.3.3 struct **clientListContext**∗ clientList_initialize ( )

Create , allocate and return a client list.

Return values

Pointer	to a freshly allocated client list or 0=Failure

6.33.3.4 int clientList_isClientAllowedToMakeAConnection (char ∗ *ip*)

6.33.3.5 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	

6.33.3.6 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

1=Banned,0=OK	

6.33.3.7 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	

6.34 src/AmmServerlib/cache/dynamic_requests.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.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:

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, char ∗verified_filename, unsigned int verified_filenameLength, unsigned int index, unsigned long ∗memSize, unsigned char ∗compressionSupported, unsigned char ∗freeContentAfterUsingIt, unsigned char ∗contentContainsPathToFileToBeStreamed)

  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 (const char ∗filename, struct AmmServer_Instance ∗instance, struct AmmServer_DynamicRequest ∗rqst)

  Save Dynamic request to a file (for debugging it)

6.34.1 Function Documentation

6.34.1.1 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	

6.34.1.2 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

6.34.1.3 char∗ dynamicRequest_serveContent (struct AmmServer_Instance ∗ *instance,* struct HTTPHeader ∗ *request,* struct AmmServer_RH_Context ∗ *shared_context,* char ∗ *verified_filename,* unsigned int *verified_filenameLength,* unsigned int *index,* unsigned long ∗ *memSize,* unsigned char ∗ *compressionSupported,* unsigned char ∗ *freeContentAfterUsingIt,* unsigned char ∗ *contentContainsPathToFileToBeStreamed*)

Handles and serves a dynamic request.

Parameters

An	AmmarServer Instance
HTTPHeader	containing the request done
Resource	Context for specific dynamic Request
The	filename that got requested , that might get rewritten
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

Bug Current implementation waits for new content , should add content double buffering to always have a valid buffer and serve it instantly , `https://github.com/AmmarkoV/AmmarServer/issues/28`

Here is the call graph for this function:

6.34.1.4 int saveDynamicRequest (const 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	

6.35 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, char ∗verified_filename, unsigned int verified_filenameLength, unsigned int index, unsigned long ∗memSize, unsigned char ∗compressionSupported, unsigned char ∗freeContentAfterUsingIt, unsigned char ∗contentContainsPathToFileToBeStreamed)

  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 (const char ∗filename, struct AmmServer_Instance ∗instance, struct AmmServer_DynamicRequest ∗rqst)

  Save Dynamic request to a file (for debugging it)

6.35.1 Detailed Description

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

Author

Ammar Qammaz (AmmarkoV)

Bug Compression should be improved

6.35.2 Function Documentation

6.35.2.1 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	

6.35.2.2 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

6.35.2.3 char∗ dynamicRequest_serveContent (struct **AmmServer_Instance** ∗ *instance,* struct **HTTPHeader** ∗ *request,* struct **AmmServer_RH_Context** ∗ *shared_context,* char ∗ *verified_filename,* unsigned int *verified_filenameLength,* unsigned int *index,* unsigned long ∗ *memSize,* unsigned char ∗ *compressionSupported,* unsigned char ∗ *freeContentAfterUsingIt,* unsigned char ∗ *contentContainsPathToFileToBeStreamed*)

Handles and serves a dynamic request.

Parameters

An	AmmarServer Instance
HTTPHeader	containing the request done
Resource	Context for specific dynamic Request
The	filename that got requested , that might get rewritten
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

Bug Current implementation waits for new content , should add content double buffering to always have a valid buffer and serve it instantly , `https://github.com/AmmarkoV/AmmarServer/issues/28`

Here is the call graph for this function:

6.35.2.4 int saveDynamicRequest (const 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	

6.36 src/AmmServerlib/cache/file_caching.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <unistd.h>
#include "../AmmServerlib.h"
#include "../server_configuration.h"
#include "file_caching.h"
#include "file_compression.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_CountMemoryUsageFreeOperation (struct AmmServer_Instance ∗instance, unsigned long freedSize)

  Tool to count total memory usage after a free operation.
- int cache_CountMemoryUsageAllocateOperation (struct AmmServer_Instance ∗instance, unsigned long allocatedSize)

  Tool to count total memory usage after a memory allocation operation.
- int cache_ChangeRequestIfTemplateRequested (struct AmmServer_Instance ∗instance, char ∗request, unsigned int maxRequest, 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_RandomizeETAG (struct AmmServer_Instance ∗instance)

  Randomize Cache-Etag prefix , this causes all subsequent hits to the cache to have a different E-Tag Prefix.
- 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, const char ∗resource, unsigned int ∗index)

  Query for a resource , and return its index.
- int cache_CreateResource (struct AmmServer_Instance ∗instance, const char ∗resource, unsigned int ∗index)
- int cache_DestroyResource (unsigned int ∗index)
- int cache_LoadResourceFromDisk (struct AmmServer_Instance ∗instance, const char ∗filename, unsigned int ∗index)
- int cache_AddFile (struct AmmServer_Instance ∗instance, const 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, const 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)

  Query to remove a resource using its 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.
- int cache_RefreshResource (struct AmmServer_Instance ∗instance, struct HTTPHeader ∗request, char ∗verified_filename, unsigned int ∗index, unsigned long ∗filesize, struct stat ∗last_modification)
- char ∗ cache_GetResource (struct AmmServer_Instance ∗instance, struct HTTPHeader ∗request, unsigned int resourceCacheID, char ∗verified_filename, unsigned int verified_filenameSize, unsigned int ∗index, unsigned long ∗filesize, struct stat ∗last_modification, unsigned char ∗compressionSupported, unsigned char ∗freeContentAfterUsingIt, unsigned char ∗serveAsRegularFile)

  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.

6.36.1 Function Documentation

6.36.1.1 int cache_AddDoNOTCacheRuleForResource (struct **AmmServer_Instance** ∗ *instance,* const 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:

6.36.1.2 int cache_AddFile (struct **AmmServer_Instance** ∗ *instance,* const 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:

6.36.1.3 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:

6.36.1.4 int cache_ChangeRequestIfTemplateRequested (struct **AmmServer_Instance** ∗ *instance,* char ∗ *request,* unsigned int *maxRequest,* 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
Maximum	size of request string
Filename	pointing to directory that contains templates

Return values

1=If	request was for a template and it got changed ,0= not changed request

6.36.1.5 int cache_CountMemoryUsageAllocateOperation (struct **AmmServer_Instance** ∗ *instance,* unsigned long *allocatedSize*)

Tool to count total memory usage after a memory allocation operation.

Parameters

An	AmmarServer Instance
Size	of memory being allocated

Bug cache_CountMemoryUsageAllocateOperation should have a mutex lock so that it is well defined on massively parallel operations

Return values

1=Ok,0=Failed	

6.36.1.6 int cache_CountMemoryUsageFreeOperation (struct **AmmServer_Instance** ∗ *instance,* unsigned long *freedSize*)

Tool to count total memory usage after a free operation.

Parameters

An	AmmarServer Instance
Size	of memory being freed

Bug cache_CountMemoryUsageFreeOperation should have a mutex lock so that it is well defined on massively parallel operations

Return values

1=Ok,0=Failed	

6.36.1.7 int cache_CreateResource (struct AmmServer_Instance ∗ *instance*, const char ∗ *resource*, unsigned int ∗ *index*)

Here is the call graph for this function:

6.36.1.8 int cache_Destroy (struct AmmServer_Instance ∗ *instance*)

Deallocate and destroy the cache of an AmmarServer instance.

Parameters

An	AmmarServer Instance

Return values

1=Ok,0=Failed	

Here is the call graph for this function:

6.36.1.9 int cache_DestroyResource (unsigned int ∗ *index*)

6.36.1.10 unsigned int cache_FindResource (struct AmmServer_Instance ∗ *instance*, const 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:

6.36.1.11 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:

6.36.1.12 char∗ cache_GetResource (struct AmmServer_Instance ∗ *instance,* struct HTTPHeader ∗ *request,* unsigned int *resourceCacheID,* char ∗ *verified_filename,* unsigned int *verified_filenameSize,* unsigned int ∗ *index,* unsigned long ∗ *filesize,* struct stat ∗ *last_modification,* unsigned char ∗ *compressionSupported,* unsigned char ∗ *freeContentAfterUsingIt,* unsigned char ∗ *serveAsRegularFile*)

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
Size	of the filename of the resource
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 This function should check filesizes/dates and refresh memory snapshots 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:

6.36.1.13 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	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

Return values

1=Ok,0=Failed	

Here is the call graph for this function:

6.36.1.14 int cache_LoadResourceFromDisk (struct **AmmServer_Instance** ∗ *instance,* const char ∗ *filename,* unsigned int ∗ *index*)

Here is the call graph for this function:

6.36.1.15 int cache_RandomizeETAG (struct **AmmServer_Instance** ∗ *instance*)

Randomize Cache-Etag prefix , this causes all subsequent hits to the cache to have a different E-Tag Prefix.

Parameters

An	AmmarServer Instance

Return values

1=Ok,0=Failed	

Here is the call graph for this function:

6.36.1.16 int cache_RefreshResource (struct **AmmServer_Instance** ∗ *instance,* struct **HTTPHeader** ∗ *request,* char ∗ *verified_filename,* unsigned int ∗ *index,* unsigned long ∗ *filesize,* struct stat ∗ *last_modification*)

6.36.1.17 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:

6.36.1.18 int cache_RemoveResource (struct **AmmServer_Instance** ∗ *instance,* unsigned int *index*)

Query to remove a resource using its index.

Parameters

An	AmmarServer Instance
Index	number of resource to be removed

Return values

1=Success,0=Failure	

Here is the call graph for this function:

6.36.1.19 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:

6.36.1.20 int freeMallocIfNeeded (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	

6.37 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 cache_CountMemoryUsageFreeOperation (struct AmmServer_Instance ∗instance, unsigned long freedSize)

Tool to count total memory usage after a free operation.

- int cache_CountMemoryUsageAllocateOperation (struct AmmServer_Instance ∗instance, unsigned long allocatedSize)

  Tool to count total memory usage after a memory allocation operation.
- 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, unsigned int maxRequest, 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, const 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, const 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, const 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_RandomizeETAG (struct AmmServer_Instance ∗instance)

  Randomize Cache-Etag prefix , this causes all subsequent hits to the cache to have a different E-Tag Prefix.
- 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_RemoveResource (struct AmmServer_Instance ∗instance, unsigned int index)

  Query to remove a resource using its index.
- 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 verified_filenameSize, unsigned int ∗index, unsigned long ∗filesize, struct stat ∗last_modification, unsigned char ∗compressionSupported, unsigned char ∗freeContentAfterUsingIt, unsigned char ∗serveAsRegularFile)

  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.

6.37.1 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

6.37.2 Function Documentation

6.37.2.1 int cache_AddDoNOTCacheRuleForResource (struct AmmServer_Instance ∗ *instance,* const 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:

6.37.2.2 int cache_AddFile (struct AmmServer_Instance ∗ *instance,* const 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:

6.37.2.3 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

1=Success,0=Failure	

Here is the call graph for this function:

6.37.2.4 int cache_ChangeRequestIfTemplateRequested (struct AmmServer_Instance ∗ *instance,* char ∗ *request,* unsigned int *maxRequest,* 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
Maximum	size of request string
Filename	pointing to directory that contains templates

Return values

1=If	request was for a template and it got changed ,0= not changed request

6.37.2.5 int cache_CountMemoryUsageAllocateOperation (struct AmmServer_Instance ∗ *instance,* unsigned long *allocatedSize*)

Tool to count total memory usage after a memory allocation operation.

Parameters

An	AmmarServer Instance
Size	of memory being allocated

Bug cache_CountMemoryUsageAllocateOperation should have a mutex lock so that it is well defined on massively parallel operations

Return values

1=Ok,0=Failed	

6.37.2.6 int cache_CountMemoryUsageFreeOperation (struct AmmServer_Instance ∗ *instance,* unsigned long *freedSize*)

Tool to count total memory usage after a free operation.

Parameters

An	AmmarServer Instance
Size	of memory being freed

Bug cache_CountMemoryUsageFreeOperation should have a mutex lock so that it is well defined on massively parallel operations

Return values

1=Ok,0=Failed	

6.37.2.7 int cache_Destroy (struct AmmServer_Instance ∗ *instance*)

Deallocate and destroy the cache of an AmmarServer instance.

Parameters

An	AmmarServer Instance

Return values

1=Ok,0=Failed	

Here is the call graph for this function:

6.37.2.8 unsigned int cache_FindResource (struct **AmmServer_Instance** ∗ *instance,* const 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:

6.37.2.9 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:

6.37.2.10 char∗ cache_GetResource (struct **AmmServer_Instance** ∗ *instance,* struct **HTTPHeader** ∗ *request,* unsigned int *resourceCacheID,* char ∗ *verified_filename,* unsigned int *verified_filenameSize,* unsigned int ∗ *index,* unsigned long ∗ *filesize,* struct stat ∗ *last_modification,* unsigned char ∗ *compressionSupported,* unsigned char ∗ *freeContentAfterUsingIt,* unsigned char ∗ *serveAsRegularFile*)

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

Size	of the filename of the resource
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 This function should check filesizes/dates and refresh memory snapshots 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:

6.37.2.11 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	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

Return values

1=Ok,0=Failed	

Here is the call graph for this function:

6.37.2.12 int cache_RandomizeETAG (struct AmmServer_Instance ∗ *instance*)

Randomize Cache-Etag prefix , this causes all subsequent hits to the cache to have a different E-Tag Prefix.

Parameters

An	AmmarServer Instance

Return values

1=Ok,0=Failed	

Here is the call graph for this function:

6.37.2.13 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:

6.37.2.14 int cache_RemoveResource (struct AmmServer_Instance ∗ *instance,* unsigned int *index*)

Query to remove a resource using its index.

Parameters

An	AmmarServer Instance
Index	number of resource to be removed

Return values

1=Success,0=Failure	

Here is the call graph for this function:

6.37.2.15 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:

6.37.2.16 int freeMallocIfNeeded (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	

6.38 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, unsigned int compression_level)

  Create compressed version of dynamic content , this should be used via the shortcut functions that control compression levels automatically.
- 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.

6.38.1 Function Documentation

6.38.1.1 int CreateCompressedVersionofCachedResource (struct AmmServer_Instance ∗ *instance,* unsigned int *index,* unsigned int *compression_level*) [inline]

Create compressed version of dynamic content , this should be used via the shortcut functions that control compression levels automatically.

Parameters

An	AmmarServer Instance
Index	of cache item
Compression	level 1-9

Return values

1=Success,0=Failure	

Here is the call graph for this function:

6.38.1.2 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:

6.38.1.3 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:

6.38.1.4 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:

6.39 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 CreateCompressedVersionofCachedResource (struct AmmServer_Instance ∗instance, unsigned int index, unsigned int compression_level)

  Create compressed version of dynamic content , this should be used via the shortcut functions that control compression levels automatically.
- 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.

6.39.1 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

6.39.2 Function Documentation

6.39.2.1 int CreateCompressedVersionofCachedResource (struct **AmmServer_Instance** * *instance,* unsigned int *index,* unsigned int *compression_level*) [inline]

Create compressed version of dynamic content , this should be used via the shortcut functions that control compression levels automatically.

Parameters

An	AmmarServer Instance
Index	of cache item
Compression	level 1-9

Return values

1=Success,0=Failure	

Here is the call graph for this function:

6.39.2.2 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:

6.39.2.3 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:

6.39.2.4 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:

6.40 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 (const 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 ∗clearItemFunction)

  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, const char ∗key, void ∗val, unsigned int valLength)

  Add a new key to hash map.
- int hashMap_AddULong (struct hashMap ∗hm, const char ∗key, unsigned long val)

  Add a new key (integer) to hash map.
- int hashMap_FindIndex (struct hashMap ∗hm, const 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, const char ∗key, void ∗payload)

  Return payload for specified key.
- int hashMap_GetULongPayload (struct hashMap ∗hm, const char ∗key, unsigned long ∗payload)

  Return numerical payload for specified key.
- int hashMap_ContainsKey (struct hashMap ∗hm, const 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, const char ∗filename)

  Save hash map to a file.
- int hashMap_LoadToFile (struct hashMap ∗hm, const char ∗filename)

  Load hash map from a file.

6.40.1 Function Documentation

6.40.1.1 int cmpHashTableItems (const void ∗ *a,* const void ∗ *b*)

6.40.1.2 unsigned long hashFunction (const 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 ^ 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..!

6.40.1.3 int hashMap_Add (struct hashMap ∗ *hm,* const 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:

6.40.1.4 int hashMap_AddULong (struct hashMap ∗ *hm,* const 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:

6.40.1.5 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:

6.40.1.6 int hashMap_ContainsKey (struct **hashMap** ∗ *hm,* const 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:

6.40.1.7 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:

6.40.1.8 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:

6.40.1.9 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:

6.40.1.10 int hashMap_FindIndex (struct hashMap ∗ *hm*, const 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:

6.40.1.11 int hashMap_GetCurrentNumberOfEntries (struct hashMap ∗ *hm*)

Get the current number of entries of hash map.

Parameters

HashMap	

Return values

Number	of entries

Here is the call graph for this function:

6.40.1.12 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:

6.40.1.13 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:

6.40.1.14 int hashMap_GetMaxNumberOfEntries (struct hashMap ∗ *hm*)

Get the maximum number of entries of hash map.

Parameters

HashMap	

Return values

Maximum	Number of entries

Here is the call graph for this function:

6.40.1.15 int hashMap_GetPayload (struct hashMap ∗ *hm*, const char ∗ *key*, void ∗ *payload*)

Return 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:

6.40.1.16 int hashMap_GetULongPayload (struct hashMap ∗ *hm*, const 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:

6.40.1.17 int hashMap_Grow (struct hashMap ∗ *hm,* unsigned int *growthSize*)

6.40.1.18 int hashMap_IsOK (struct hashMap ∗ *hm*)

6.40.1.19 int hashMap_IsSorted (struct hashMap ∗ *hm*)

Here is the call graph for this function:

6.40.1.20 int hashMap_LoadToFile (struct hashMap ∗ *hm,* const char ∗ *filename*)

Load hash map from a file.

Parameters

HashMap	structure
Filename	to save to

Return values

1=Success,0=Fail	

6.40.1.21 int hashMap_SaveToFile (struct hashMap ∗ *hm,* const char ∗ *filename*)

Save hash map to a file.

Parameters

HashMap	structure
Filename	to save to

Return values

1=Success,0=Fail	

Here is the call graph for this function:

6.40.1.22 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:

6.40.1.23 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

Return values

1=Success,0=Failure	

Here is the call graph for this function:

6.41 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.

Macros

- #define HASHMAP_BE_THREAD_SAFE 1

  HashMap should always be thread safe , since we are talking about a multi-threaded web-server. That being said , if someone wants to use hashmap.c/hashmap.h as a standalone ingredient to another project and wants to discard all thread specific locks , it can be easily done with the following switch..!

Functions

- unsigned long hashFunction (const 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 ∗clearItemFunction)

  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, const char ∗key, void ∗val, unsigned int valLength)

  Add a new key to hash map.
- int hashMap_AddULong (struct hashMap ∗hm, const char ∗key, unsigned long val)

  Add a new key (integer) to hash map.
- int hashMap_FindIndex (struct hashMap ∗hm, const 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, const char ∗key, void ∗payload)

  Return payload for specified key.
- int hashMap_GetULongPayload (struct hashMap ∗hm, const 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, const 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, const char ∗filename)

  Load hash map from a file.
- int hashMap_SaveToFile (struct hashMap ∗hm, const char ∗filename)

  Save hash map to a file.

6.41.1 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

6.41.2 Macro Definition Documentation

6.41.2.1 #define HASHMAP_BE_THREAD_SAFE 1

HashMap should always be thread safe , since we are talking about a multi-threaded web-server. That being said , if someone wants to use hashmap.c/hashmap.h as a standalone ingredient to another project and wants to discard all thread specific locks , it can be easily done with the following switch..!

6.41.3 Function Documentation

6.41.3.1 unsigned long hashFunction (const 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 ^ 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..!

6.41.3.2 int hashMap_Add (struct hashMap ∗ *hm,* const 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:

6.41.3.3 int hashMap_AddULong (struct hashMap ∗ *hm,* const 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:

6.41.3.4 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:

6.41.3.5 int hashMap_ContainsKey (struct hashMap * *hm*, const 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:

6.41.3.6 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:

6.41.3.7 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:

6.41.3.8 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:

6.41.3.9 int hashMap_FindIndex (struct hashMap ∗ *hm,* const 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:

6.41.3.10 int hashMap_GetCurrentNumberOfEntries (struct hashMap ∗ *hm*)

Get the current number of entries of hash map.

Parameters

HashMap	

Return values

Number	of entries

Here is the call graph for this function:

6.41.3.11 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:

6.41.3.12 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:

6.41.3.13 int hashMap_GetMaxNumberOfEntries (struct hashMap ∗ *hm*)

Get the maximum number of entries of hash map.

Parameters

HashMap	

Return values

Maximum	Number of entries

Here is the call graph for this function:

6.41.3.14 int hashMap_GetPayload (struct hashMap ∗ *hm,* const char ∗ *key,* void ∗ *payload*)

Return 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:

6.41.3.15 int hashMap_GetULongPayload (struct hashMap ∗ *hm,* const 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:

6.41.3.16 int hashMap_LoadToFile (struct hashMap ∗ *hm,* const char ∗ *filename*)

Load hash map from a file.

Parameters

HashMap	structure
Filename	to save to

Return values

1=Success,0=Fail	

6.41.3.17 int hashMap_SaveToFile (struct hashMap ∗ *hm,* const char ∗ *filename*)

Save hash map to a file.

Parameters

HashMap	structure
Filename	to save to

Return values

1=Success,0=Fail	

Here is the call graph for this function:

6.41.3.18 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:

6.41.3.19 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

Return values

1=Success,0=Failure	

Here is the call graph for this function:

6.42 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 <sys/types.h>
#include <sys/socket.h>
#include <stdio.h>
#include <stdlib.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 ∗headerLength)

  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)

6.42.1 Macro Definition Documentation

6.42.1.1 #define CR 13

6.42.1.2 #define LF 10

6.42.2 Function Documentation

6.42.2.1 int AnalyzeHTTPHeader (struct AmmServer_Instance ∗ *instance,* struct HTTPTransaction ∗ *transaction*)

Analyze HTTP header (after it has been accumulated into memory)

Parameters

An	AmmarServer Instance
HTTPTransaction	we are talking about

Return values

1=Success,0=Failure	

Here is the call graph for this function:

6.42.2.2 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:

6.42.2.3 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:

6.42.2.4 int FreeHTTPHeader (struct **HTTPHeader** ∗ *output*)

Deallocate memory occupied by an HTTP Header.

Parameters

HTTPHeader	to be deallocated

Return values

1=Success,0=Failure	

6.42.2.5 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	

6.42.2.6 int HTTPHeaderIsPOST (char ∗ *request,* unsigned int *requestLength*)

Ask if a header is a POST request, detected by the first four consecutive bytes being P O S T.

Parameters

Pointer	to incoming request (streaming) string
Length	of incoming string

Return values

1=POST,0=Not	POST request

6.42.2.7 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:

6.42.2.8 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:

6.42.2.9 int ProcessRangeHTTPLine (char ∗ *request*, unsigned int *requestLength*, unsigned long ∗ *rangeStart*, unsigned long ∗ *rangeEnd*) [inline]

Bug : ProcessRangeHTTPLine , can be improved , it is not thoroughly tested

Here is the call graph for this function:

6.42.2.10 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

Bug Reallocation code of ReceiveHTTPHeader when we jump from a regular GET memory block to a large POST memory block is shit and needs to be fixed

Here is the call graph for this function:

6.43 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 ∗headerLength)

  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)

6.43.1 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

6.43.2 Function Documentation

6.43.2.1 int AnalyzeHTTPHeader (struct AmmServer_Instance ∗ *instance,* struct HTTPTransaction ∗ *transaction*)

Analyze HTTP header (after it has been accumulated into memory)

Parameters

An	AmmarServer Instance
HTTPTransaction	we are talking about

Return values

1=Success,0=Failure	

Here is the call graph for this function:

6.43.2.2 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:

6.43.2.3 int FreeHTTPHeader (struct **HTTPHeader** ∗ *output*)

Deallocate memory occupied by an HTTP Header.

Parameters

HTTPHeader	to be deallocated

Return values

1=Success,0=Failure	

6.43.2.4 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	

6.43.2.5 int HTTPHeaderIsPOST (char ∗ *request,* unsigned int *requestLength*)

Ask if a header is a POST request, detected by the first four consecutive bytes being P O S T.

Parameters

Pointer	to incoming request (streaming) string
Length	of incoming string

Return values

1=POST,0=Not	POST request

6.43.2.6 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

Return values

Pointer	to memory containing HTTPHeader,0=Failure

Bug Reallocation code of ReceiveHTTPHeader when we jump from a regular GET memory block to a large POST memory block is shit and needs to be fixed

Here is the call graph for this function:

6.44 src/AmmServerlib/header_analysis/post_header_analysis.c File Reference

```
#include "post_header_analysis.h"
#include "../tools/logs.h"
#include "../tools/http_tools.h"
#include "../stringscanners/postHeader.h"
#include "../stringscanners/httpHeader.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)

  Analyze a POST request line by line filling in the structures that define it.

6.44.1 Function Documentation

6.44.1.1 int AnalyzePOSTLineRequest (struct AmmServer_Instance ∗ *instance,* struct HTTPHeader ∗ *output,* char ∗ *request,* unsigned int *request_length,* unsigned int *lines_gathered*)

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:

6.45 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)

  Analyze a POST request line by line filling in the structures that define it.

6.45.1 Detailed Description

Tools to process POST requests.

Author

Ammar Qammaz (AmmarkoV)

Bug POST header analysis is not fully implemented yet

6.45.2 Function Documentation

6.45.2.1 int AnalyzePOSTLineRequest (struct **AmmServer_Instance** ∗ *instance,* struct **HTTPHeader** ∗ *output,* char ∗ *request,* unsigned int *request_length,* unsigned int *lines_gathered*)

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:

6.46 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"

6.46.1 Function Documentation

6.46.1.1 const char∗ Version ( )

Here is the call graph for this function:

6.46.2 Variable Documentation

6.46.2.1 const char∗ ver =" VERSION 1.30 - 7/1/10 \0"

6.47 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

6.48 src/AmmServerlib/InputParser/InputParser_C.c File Reference

```
#include "InputParser_C.h"
#include <math.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 [] =" 0.357 written from scratch - 8/2/10 \0"

6.48.1 Macro Definition Documentation

6.48.1.1 #define WARN_ABOUT_INCORRECTLY_ALLOCATED_STACK_STRINGS 1

6.48.2 Function Documentation

6.48.2.1 unsigned char CheckDelimeterNumOk (struct InputParserC ∗ *ipc,* int *num*) [inline]

6.48.2.2 unsigned char CheckIPCOk (struct InputParserC ∗ *ipc*) [inline]

6.48.2.3 unsigned char CheckWordNumOk (struct InputParserC ∗ *ipc,* unsigned int *num*) [inline]

Here is the call graph for this function:

6.48.2.4 int InputParser_ClearNonCharacters (char ∗ *inpt,* unsigned int *length*)

6.48.2.5 struct InputParserC∗ InputParser_Create (unsigned int *max_string_count,* unsigned int *max_delimiter_count*)

Here is the call graph for this function:

6.48.2.6 void InputParser_DefaultDelimeters (struct InputParserC ∗ *ipc*)

Here is the call graph for this function:

6.48.2.7 void InputParser_Destroy (struct InputParserC ∗ *ipc*)

6.48.2.8 char InputParser_GetDelimeter (struct InputParserC ∗ *ipc,* int *num*)

Here is the call graph for this function:

6.48.2.9 unsigned int InputParser_GetLowercaseWord (struct InputParserC ∗ *ipc,* unsigned int *num,* char ∗ *wheretostore,* unsigned int *storagesize*)

Here is the call graph for this function:

6.48.2.10 unsigned int InputParser_GetUpcaseWord (struct InputParserC ∗ *ipc,* unsigned int *num,* char ∗ *wheretostore,* unsigned int *storagesize*)

Here is the call graph for this function:

6.48.2.11 unsigned int InputParser_GetWord (struct InputParserC ∗ *ipc,* unsigned int *num,* char ∗ *wheretostore,* unsigned int *storagesize*)

Here is the call graph for this function:

6.48.2.12 char InputParser_GetWordChar (struct InputParserC ∗ *ipc,* unsigned int *num,* unsigned int *pos*)

Here is the call graph for this function:

6.48.2.13 float InputParser_GetWordFloat (struct InputParserC ∗ *ipc,* unsigned int *num*)

Here is the call graph for this function:

6.48.2.14 signed int InputParser_GetWordInt (struct InputParserC ∗ *ipc,* unsigned int *num*)

Here is the call graph for this function:

6.48.2.15 unsigned int InputParser_GetWordLength (struct InputParserC ∗ *ipc,* unsigned int *num*)

Here is the call graph for this function:

6.48.2.16 unsigned char InputParser_SelfCheck (struct InputParserC ∗ *ipc*)

Here is the call graph for this function:

6.48.2.17 int InputParser_SeperateWords (struct InputParserC ∗ *ipc,* char ∗ *inpt,* char *keepcopy*)

Here is the call graph for this function:

6.48.2.18 int InputParser_SeperateWordsCC (struct InputParserC ∗ *ipc,* const char ∗ *inpt,* char *keepcopy*)

Here is the call graph for this function:

6.48.2.19 int InputParser_SeperateWordsUC (struct InputParserC ∗ *ipc,* unsigned char ∗ *inpt,* char *keepcopy*)

Here is the call graph for this function:

6.48.2.20 void InputParser_SetDelimeter (struct InputParserC ∗ *ipc,* int *num,* char *tmp*)

Here is the call graph for this function:

6.48.2.21 int InputParser_TrimCharacters (char ∗ *inpt,* unsigned int *length,* char *what2trim*)

Here is the call graph for this function:

6.48.2.22 int InputParser_TrimCharactersEnd (char ∗ *inpt,* unsigned int *length,* char *what2trim*)

6.48.2.23 int InputParser_TrimCharactersStart (char ∗ *inpt,* unsigned int *length,* char *what2trim*)

6.48.2.24 unsigned char InputParser_WordCompare (struct InputParserC ∗ *ipc,* unsigned int *num,* char ∗ *word,* unsigned int *wordsize*)

Here is the call graph for this function:

6.48.2.25 unsigned char InputParser_WordCompareAuto (struct InputParserC ∗ *ipc,* unsigned int *num,* char ∗ *word*)

Here is the call graph for this function:

6.48.2.26 unsigned char InputParser_WordCompareNoCase (struct InputParserC ∗ *ipc,* unsigned int *num,* char ∗ *word,* unsigned int *wordsize*)

Here is the call graph for this function:

6.48.2.27 unsigned char InputParser_WordCompareNoCaseAuto (struct InputParserC ∗ *ipc,* unsigned int *num,* char ∗ *word*)

Here is the call graph for this function:

6.48.2.28 char∗ InputParserC_Version ( )

6.48.2.29 signed int Str2Int_internal (char ∗ *inpt,* unsigned int *start_from,* unsigned int *length*) [inline]

6.48.3 Variable Documentation

6.48.3.1 char _ipc_ver[ ] =" 0.357 written from scratch - 8/2/10 \0"

6.48.3.2 int warningsAboutIncorrectlyAllocatedStackIssued = 0

6.49 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
- #define USE_SCANF 0

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)

6.49.1 Macro Definition Documentation

6.49.1.1 #define CONTAINERS_MAX 1

6.49.1.2 #define DELIM_MAX_MAX 6

6.49.1.3 #define MAX_COMPLICITY 4

6.49.1.4 #define MAX_MEMORY 256

6.49.1.5 #define MAX_STRING 2048

6.49.1.6 #define USE_SCANF 0

6.49.2 Function Documentation

6.49.2.1 unsigned char CheckWordNumOk (struct InputParserC ∗ *ipc,* unsigned int *num*) [inline]

Here is the call graph for this function:

6.49.2.2 int InputParser_ClearNonCharacters (char ∗ *inpt,* unsigned int *length*)

6.49.2.3 struct InputParserC∗ InputParser_Create (unsigned int *max_string_count,* unsigned int *max_delimiter_count*)

Here is the call graph for this function:

6.49.2.4 void InputParser_DefaultDelimeters (struct InputParserC ∗ *ipc*)

Here is the call graph for this function:

6.49.2.5 void InputParser_Destroy (struct InputParserC ∗ *ipc*)

6.49.2.6 char InputParser_GetDelimeter (struct InputParserC ∗ *ipc,* int *num*)

Here is the call graph for this function:

6.49.2.7 unsigned int InputParser_GetLowercaseWord (struct InputParserC ∗ *ipc,* unsigned int *num,* char ∗ *wheretostore,* unsigned int *storagesize*)

Here is the call graph for this function:

6.49.2.8 unsigned int InputParser_GetUpcaseWord (struct InputParserC ∗ *ipc,* unsigned int *num,* char ∗ *wheretostore,* unsigned int *storagesize*)

Here is the call graph for this function:

6.49.2.9 unsigned int InputParser_GetWord (struct InputParserC ∗ *ipc,* unsigned int *num,* char ∗ *wheretostore,* unsigned int *storagesize*)

Here is the call graph for this function:

6.49.2.10 char InputParser_GetWordChar (struct InputParserC ∗ *ipc,* unsigned int *num,* unsigned int *pos*)

Here is the call graph for this function:

6.49.2.11 float InputParser_GetWordFloat (struct InputParserC ∗ *ipc,* unsigned int *num*)

Here is the call graph for this function:

6.49.2.12 signed int InputParser_GetWordInt (struct InputParserC ∗ *ipc,* unsigned int *num*)

Here is the call graph for this function:

6.49.2.13 unsigned int InputParser_GetWordLength (struct InputParserC ∗ *ipc,* unsigned int *num*)

Here is the call graph for this function:

6.49.2.14 unsigned char InputParser_SelfCheck (struct InputParserC ∗ *ipc*)

Here is the call graph for this function:

6.49.2.15 int InputParser_SeperateWords (struct InputParserC ∗ *ipc,* char ∗ *inpt,* char *keepcopy*)

Here is the call graph for this function:

6.49.2.16 int InputParser_SeperateWordsCC (struct InputParserC ∗ *ipc,* const char ∗ *inpt,* char *keepcopy*)

Here is the call graph for this function:

6.49.2.17 int InputParser_SeperateWordsUC (struct InputParserC ∗ *ipc,* unsigned char ∗ *inpt,* char *keepcopy*)

Here is the call graph for this function:

6.49.2.18 void InputParser_SetDelimeter (struct InputParserC ∗ *ipc,* int *num,* char *tmp*)

Here is the call graph for this function:

6.49.2.19 int InputParser_TrimCharacters (char ∗ *inpt,* unsigned int *length,* char *what2trim*)

Here is the call graph for this function:

6.49.2.20 int InputParser_TrimCharactersEnd (char ∗ *inpt,* unsigned int *length,* char *what2trim*)

6.49.2.21 int InputParser_TrimCharactersStart (char ∗ *inpt,* unsigned int *length,* char *what2trim*)

6.49.2.22 unsigned char InputParser_WordCompare (struct InputParserC ∗ *ipc,* unsigned int *num,* char ∗ *word,* unsigned int *wordsize*)

Here is the call graph for this function:

6.49.2.23 unsigned char InputParser_WordCompareAuto (struct **InputParserC** ∗ *ipc,* unsigned int *num,* char ∗ *word*)

Here is the call graph for this function:

6.49.2.24 unsigned char InputParser_WordCompareNoCase (struct **InputParserC** ∗ *ipc,* unsigned int *num,* char ∗ *word,* unsigned int *wordsize*)

Here is the call graph for this function:

6.49.2.25 unsigned char InputParser_WordCompareNoCaseAuto (struct **InputParserC** ∗ *ipc,* unsigned int *num,* char ∗ *word*)

Here is the call graph for this function:

6.49.2.26 char∗ InputParserC_Version ( )

6.50 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

6.50.1 Function Documentation

6.50.1.1 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-Transaction	this send file is part of
Error	Code to send

Return values

1=Success,0=Failure	

Here is the call graph for this function:

6.50.1.2 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-Transaction	this send file is part of
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:

6.50.1.3 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
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:

6.50.1.4 int SendPart (int *clientsock,* char ∗ *message,* unsigned int *message_size*)

6.50.1.5 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:

6.50.1.6 int TransmitFileToSocketInternal (FILE ∗ *pFile,* int *clientsock,* unsigned long *bytesToSendStart*) [inline]

Here is the call graph for this function:

6.50.2 Variable Documentation

6.50.2.1 unsigned int files_open = 0

6.51 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.

6.51.1 Detailed Description

Basic file server functionality of AmmarServer.

Author

Ammar Qammaz (AmmarkoV)

6.51.2 Function Documentation

6.51.2.1 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-Transaction	this send file is part of
Error	Code to send

Return values

1=Success,0=Failure	

Here is the call graph for this function:

6.51.2.2 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-Transaction	this send file is part of
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:

6.51.2.3 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
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:

6.52 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 "../tools/http_tools.h"
#include "../tools/time_provider.h"
```

Include dependency graph for sendHTTPHeader.c:

Functions

- unsigned long SendErrorCodeHeader (int clientsock, unsigned int error_code, const char ∗verified_filename, const char ∗templates_root)

  Send an Error Code header.
- unsigned long SendSuccessCodeHeader (int clientsock, int success_code, const 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, const char ∗verified_filename)

  Send a 401 Not Authorized response.

6.52.1 Function Documentation

6.52.1.1 unsigned long SendAuthorizationHeader (int *clientsock,* char ∗ *message,* const 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:

6.52.1.2 unsigned long SendErrorCodeHeader (int *clientsock,* unsigned int *error_code,* const char ∗ *verified_filename,* const 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

Bug This call seems to fail ?

Return values

1=Success,0=Failure	

Here is the call graph for this function:

6.52.1.3 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:

6.52.1.4 unsigned long SendSuccessCodeHeader (int *clientsock,* int *success_code,* const 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:

6.53 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, const char ∗verified_filename, const char ∗templates_root)

  Send an Error Code header.
- unsigned long SendSuccessCodeHeader (int clientsock, int success_code, const 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, const char ∗verified_filename)

  Send a 401 Not Authorized response.

6.53.1 Detailed Description

Small code segments that transmit HTTP responses.

Author

Ammar Qammaz (AmmarkoV)

6.53.2 Function Documentation

6.53.2.1 unsigned long SendAuthorizationHeader (int *clientsock,* char ∗ *message,* const 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:

6.53.2.2 unsigned long SendErrorCodeHeader (int *clientsock,* unsigned int *error_code,* const char ∗ *verified_filename,* const 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

Bug This call seems to fail ?

Return values

1=Success,0=Failure	

Here is the call graph for this function:

6.53.2.3 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:

6.53.2.4 unsigned long SendSuccessCodeHeader (int *clientsock,* int *success_code,* const 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:

6.54 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, const char ∗conf_file)
- int AssignStr (char ∗∗dest, const 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 =1
- char AccessLog [MAX_FILE_PATH] ="access.log"
- int ErrorLogEnable =1
- char ErrorLog [MAX_FILE_PATH] ="error.log"
- char TemplatesInternalURI [MAX_RESOURCE] = TEMPLATE_INTERNAL_URI

6.54.1 Function Documentation

6.54.1.1 int AssignStr (char ∗∗ *dest,* const char ∗ *source*)

6.54.1.2 int EmmitPossibleConfigurationWarnings (struct AmmServer_Instance ∗ *instance*)

Bug TOP PRIORITY -> Implement POST !FILE! requests , and couple them to dynamic content

Bug Implement download resume capabilities (range head request) ..

Bug require the Host: header from HTTP 1.1 clients

Bug accept absolute URL's in a request

Bug accept requests with chunked data

Bug use the "100 Continue" response appropriately

Bug handle requests with If-Modified-Since: or If-Unmodified-Since: headers

Bug Add configuration file ammServ.conf parsing..

Bug Add detailed input header parsing

Bug Improve directory listings (add filesizes , dates etc)

Bug Improve implemented file caching mechanism (add string comparison to make code hash collision free)

Bug Add apache like logging capabilities

Here is the call graph for this function:

6.54.1.3 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	

6.54.1.4 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	

6.54.1.5 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	

6.54.1.6 int LoadConfigurationFile (struct AmmServer_Instance ∗ *instance,* const 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:

6.54.1.7 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:

6.54.2 Variable Documentation

6.54.2.1 char AccessLog[MAX_FILE_PATH] ="access.log"

6.54.2.2 int AccessLogEnable =1

6.54.2.3 unsigned char CACHING_ENABLED =1

If caching is disabled server becomes a very simple file server , dynamic requests are also disabled.

6.54.2.4 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)

6.54.2.5 int CHANGE_TO_UID =NON_ROOT_UID_IF_USER_FAILS

6.54.2.6 char ErrorLog[MAX_FILE_PATH] ="error.log"

6.54.2.7 int ErrorLogEnable =1

6.54.2.8 unsigned int GLOBAL_KILL_SERVER_SWITCH = 0

Setting this to 1 will signal that all instances of AmmarServer need to die at once.

6.54.2.9 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)

6.54.2.10 int MAX_CACHE_SIZE_IN_MB = 128

Maximum memory usage (Megabytes) for the entire cache (per instance of AmmarServer)

6.54.2.11 int MAX_SEPERATE_CACHE_ITEMS = 1024

Maximum Number of separate items in cache (per instance of AmmarServer)

6.54.2.12 char TemplatesInternalURI[MAX_RESOURCE] = TEMPLATE_INTERNAL_URI

6.54.2.13 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.

6.54.2.14 int varSocketTimeoutREAD_seconds = DEFAULT_SOCKET_READ_TIMEOUT_SECS

6.54.2.15 int varSocketTimeoutWRITE_seconds =DEFAULT_SOCKET_WRITE_TIMEOUT_SECS

6.55 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 WORKAROUND_REALLOCATION_R_X86_64_PC32_GCC_ERROR 1

  Redeclares a function that causes linking problems..
- #define CLIENT_SLEEP_TIME_WHEN_DYNAMIC_REQUEST_CALLBACK_IS_BUSY_NSEC 1500000

  Time sleeping when a dynamic request that serves a common file across all clients is busy.
- #define CLIENT_SLEEP_TIME_INTERVAL_NSEC 10000

  Time sleeping when a dynamic request that serves a common file across all clients is busy.
- #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_MAXIMUM_TIME_TO_WAIT_FOR_A_NEWLY_CREATED_THREAD_MS 250000

  Max sleep time while waiting for new thread to kick in and read parameters to unblock main thread..
- #define THREAD_SLEEP_TIME_WHILE_WAITING_FOR_NEW_CREATED_THREAD_TO_CONSUME_PARAMETERS 20

  Sleep time while waiting for new thread to kick in and read parameters to unblock main thread..
- #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 DELAY_TRY_BINDING_TO_PORT 5000 ∗1000

  Sleep time after unsuccessfully trying to bind to port (usleep(DELAY...)
- #define MAX_TRIES_TO_BIND_TO_PORT 5

  Maximum times to try to bind to port on initial server start up.
- #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.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 RANDOMIZE_ETAG_PER_LAUNCH 1

  This enables e-tag randomization on each creation of a cache , this makes clients automatically refresh when server is restarted.
- #define MAX_ETAG_SIZE 128

  Maximum size of an E-Tag.
- #define MAX_HTTP_REQUEST_HEADER_REPLY 1024

  Maximum size of an http header reply.
- #define MAX_HTTP_REQUEST_SHORT_HEADER_REPLY 512

  Maximum size of a short , static , 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 256/∗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 5

  Default timeout value before which a socket blocking on a read call should be considered dead.
- #define DEFAULT_SOCKET_WRITE_TIMEOUT_SECS 5

  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, const char ∗conf_file)

- int AssignStr (char ∗∗dest, const 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]

6.55.1 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

6.55.2 Macro Definition Documentation

6.55.2.1 #define CALCULATE_TIME_FOR_UPLOADS 1

Calculate (And output) transmission speed for files broadcast by AmmarServer.

6.55.2.2 #define CLIENT_SLEEP_TIME_INTERVAL_NSEC 10000

Time sleeping when a dynamic request that serves a common file across all clients is busy.

6.55.2.3 #define CLIENT_SLEEP_TIME_WHEN_DYNAMIC_REQUEST_CALLBACK_IS_BUSY_NSEC 1500000

Time sleeping when a dynamic request that serves a common file across all clients is busy.

6.55.2.4 #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.

6.55.2.5 #define DEFAULT_SOCKET_READ_TIMEOUT_SECS 5

Default timeout value before which a socket blocking on a read call should be considered dead.

6.55.2.6 #define DEFAULT_SOCKET_WRITE_TIMEOUT_SECS 5

Default timeout value before which a socket blocking on a write call should be considered dead.

6.55.2.7 #define DEFAULT_USERNAME_UID_FOR_DAEMON "www-data"

Default Username to change to if we are running from root.

6.55.2.8 #define DELAY_TRY_BINDING_TO_PORT 5000 $*$1000

Sleep time after unsuccessfully trying to bind to port (usleep(DELAY...)

6.55.2.9 #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.

6.55.2.10 #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.

6.55.2.11 #define ENABLE_DIRECTORY_LISTING 1

Enable directory listing , if this is disabled attack surface gets significantly reduced.

6.55.2.12 #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.

6.55.2.13 #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)

6.55.2.14 #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.

6.55.2.15 #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)

6.55.2.16 #define ENABLE_POST 1

Enable POST request handling , switching this to 0 will completely deny them reducing attack surface.

6.55.2.17 #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.

6.55.2.18 #define GROWSTEP_DIRECTORY_LIST_RESPONSE_BODY 16/*KB*/*1024

Controls allocation step for when we run out of space for a directory listing.

6.55.2.19 #define HTTP_POST_GROWTH_STEP_REQUEST_HEADER 512/*KB*/*1024

Maximum size of an incoming HTTP Header allocation step.

6.55.2.20 #define INITIAL_DIRECTORY_LIST_RESPONSE_BODY 64/*KB*/*1024

Controls initial allocated size for a directory listing.

6.55.2.21 #define MAX_CLIENT_PRESPAWNED_THREADS 0

Prespawned theads reduce overall latency but they increase CPU load , 0 disables them.

6.55.2.22 #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.

6.55.2.23 #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`)

6.55.2.24 #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

6.55.2.25 #define MAX_CONFIGURATION_FILE_LINE_SIZE 512

Maximum line length in configuration file.

6.55.2.26 #define MAX_CONTENT_TYPE 128

Maximum length of a content type record.

6.55.2.27 #define MAX_DIRECTORY_LIST_RESPONSE_BODY 256/∗KB∗/∗1024

Maximum space allocated for a directory listing.

6.55.2.28 #define MAX_ETAG_SIZE 128

Maximum size of an E-Tag.

6.55.2.29 #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.

6.55.2.30 #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)

6.55.2.31 #define MAX_HTTP_REQUEST_HEADER 4096

Maximum size of an incoming HTTP Header.

6.55.2.32 #define MAX_HTTP_REQUEST_HEADER_LINES 1024

An incoming header should not have more than X numbers of lines.

6.55.2.33 #define MAX_HTTP_REQUEST_HEADER_REPLY 1024

Maximum size of an http header reply.

6.55.2.34 #define MAX_HTTP_REQUEST_SHORT_HEADER_REPLY 512

Maximum size of a short , static , http header reply.

6.55.2.35 #define MAX_RESOURCE_SLASHES 15

Max slashes in a Resource (i.e. `http://xxx.xxx.xxx.xxx/test/resource` has 4 slashes.

6.55.2.36 #define MAX_TRIES_TO_BIND_TO_PORT 5

Maximum times to try to bind to port on initial server start up.

6.55.2.37 #define NON_ROOT_UID_IF_USER_FAILS 1500

Non Root UID to change to.

6.55.2.38 #define RANDOMIZE_ETAG_PER_LAUNCH 1

This enables e-tag randomization on each creation of a cache , this makes clients automatically refresh when server is restarted.

6.55.2.39 #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.

6.55.2.40 #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

6.55.2.41 #define THREAD_MAXIMUM_TIME_TO_WAIT_FOR_A_NEWLY_CREATED_THREAD_MS 250000

Max sleep time while waiting for new thread to kick in and read parameters to unblock main thread..

6.55.2.42 #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)

6.55.2.43 #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)

6.55.2.44 #define THREAD_SLEEP_TIME_WHILE_WAITING_FOR_NEW_CREATED_THREAD_TO_CONSUME_PARAMETERS 20

Sleep time while waiting for new thread to kick in and read parameters to unblock main thread..

6.55.2.45 #define WORKAROUND_REALLOCATION_R_X86_64_PC32_GCC_ERROR 1

Redeclares a function that causes linking problems..

6.55.3 Function Documentation

6.55.3.1 int AssignStr (char ∗∗ *dest,* const char ∗ *source*)

6.55.3.2 int EmmitPossibleConfigurationWarnings ( )

Internal check of server configuration and possible error messages in impossible situations.

Return values

1=Success,0=Failure	

6.55.3.3 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	

6.55.3.4 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	

6.55.3.5 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=Ok,0=Don'tAllocate	

6.55.3.6 int LoadConfigurationFile (struct AmmServer_Instance ∗ *instance,* const 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:

6.55.3.7 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:

6.55.4 Variable Documentation

6.55.4.1 char AccessLog[MAX_FILE_PATH]

6.55.4.2 int AccessLogEnable

6.55.4.3 unsigned char CACHING_ENABLED

If caching is disabled server becomes a very simple file server , dynamic requests are also disabled.

6.55.4.4 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)

6.55.4.5 int CHANGE_TO_UID

6.55.4.6 char ErrorLog[MAX_FILE_PATH]

6.55.4.7 int ErrorLogEnable

6.55.4.8 unsigned int GLOBAL_KILL_SERVER_SWITCH

Setting this to 1 will signal that all instances of AmmarServer need to die at once.

6.55.4.9 int MAX_CACHE_SIZE_FOR_EACH_FILE_IN_MB

Maximum memory usage (Megabytes) for a specific entry of the cache (per instance of AmmarServer)

6.55.4.10 int MAX_CACHE_SIZE_IN_MB

Maximum memory usage (Megabytes) for the entire cache (per instance of AmmarServer)

6.55.4.11 int MAX_SEPERATE_CACHE_ITEMS

Maximum Number of separate items in cache (per instance of AmmarServer)

6.55.4.12 char TemplatesInternalURI[MAX_RESOURCE]

6.55.4.13 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.

6.55.4.14 int varSocketTimeoutREAD_seconds

6.55.4.15 int varSocketTimeoutWRITE_seconds

6.56 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 (const char ∗str, unsigned int strLength)

    Scan a string for one of the words of the applicationFiles word set.

6.56.1 Function Documentation

6.56.1.1 int scanFor_applicationFiles (const 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

6.57 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 (const char ∗str, unsigned int strLength)

  Scan a string for one of the words of the applicationFiles word set.

6.57.1 Detailed Description

A tool that scans for a string in a very fast and robust way.

Author

Ammar Qammaz (AmmarkoV)

6.57.2 Enumeration Type Documentation

6.57.2.1 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

6.57.3 Function Documentation

6.57.3.1 int scanFor_applicationFiles (const 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

6.58 src/AmmServerlib/stringscanners/archiveFiles.c File Reference

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>
#include "archiveFiles.h"
```

Include dependency graph for archiveFiles.c:

Functions

- int scanFor_archiveFiles (const char ∗str, unsigned int strLength)

  Scan a string for one of the words of the archiveFiles word set.

6.58.1 Function Documentation

6.58.1.1 int scanFor_archiveFiles (const char ∗ *str,* unsigned int *strLength*)

Scan a string for one of the words of the archiveFiles word set.

Parameters

Input	String , to be scanned
Length	of Input String

Return values

See	above enumerator

6.59 src/AmmServerlib/stringscanners/archiveFiles.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 {
  ARCHIVEFILES_EMPTY =0, ARCHIVEFILES_7Z, ARCHIVEFILES_AR, ARCHIVEFILES_BZ2,
  ARCHIVEFILES_CBZ, ARCHIVEFILES_CPIO, ARCHIVEFILES_GZ, ARCHIVEFILES_ISO,
  ARCHIVEFILES_JAR, ARCHIVEFILES_LZMA, ARCHIVEFILES_TAR, ARCHIVEFILES_TGZ,
  ARCHIVEFILES_TAR_7Z, ARCHIVEFILES_TAR_Z, ARCHIVEFILES_TAR_GZ, ARCHIVEFILES_TAR_BZ2,
  ARCHIVEFILES_TAR_BZ, ARCHIVEFILES_TAR_LZ, ARCHIVEFILES_TAR_LZMA, ARCHIVEFILES_TAR_XZ,
  ARCHIVEFILES_XZ, ARCHIVEFILES_ZIP, ARCHIVEFILES_END_OF_ITEMS }

Enumerator for the IDs of archiveFiles so we can know what the result was.

Functions

- int scanFor_archiveFiles (const char ∗str, unsigned int strLength)

  Scan a string for one of the words of the archiveFiles word set.

6.59.1 Detailed Description

A tool that scans for a string in a very fast and robust way.

Author

Ammar Qammaz (AmmarkoV)

6.59.2 Enumeration Type Documentation

6.59.2.1 anonymous enum

Enumerator for the IDs of archiveFiles so we can know what the result was.

Enumerator

ARCHIVEFILES_EMPTY

ARCHIVEFILES_7Z

ARCHIVEFILES_AR

ARCHIVEFILES_BZ2

ARCHIVEFILES_CBZ

ARCHIVEFILES_CPIO

ARCHIVEFILES_GZ

ARCHIVEFILES_ISO

ARCHIVEFILES_JAR

ARCHIVEFILES_LZMA

ARCHIVEFILES_TAR

ARCHIVEFILES_TGZ

ARCHIVEFILES_TAR_7Z

ARCHIVEFILES_TAR_Z

ARCHIVEFILES_TAR_GZ

ARCHIVEFILES_TAR_BZ2

ARCHIVEFILES_TAR_BZ

ARCHIVEFILES_TAR_LZ

ARCHIVEFILES_TAR_LZMA

ARCHIVEFILES_TAR_XZ

ARCHIVEFILES_XZ

ARCHIVEFILES_ZIP

ARCHIVEFILES_END_OF_ITEMS

6.59.3 Function Documentation

6.59.3.1 int scanFor_archiveFiles (const char ∗ *str,* unsigned int *strLength*)

Scan a string for one of the words of the archiveFiles word set.

Parameters

Input	String , to be scanned
Length	of Input String

Return values

See	above enumerator

6.60 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 (const char ∗str, unsigned int strLength)

  Scan a string for one of the words of the audioFiles word set.

6.60.1 Function Documentation

6.60.1.1 int scanFor_audioFiles (const 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

6.61 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 (const char ∗str, unsigned int strLength)

Scan a string for one of the words of the audioFiles word set.

6.61.1 Detailed Description

A tool that scans for a string in a very fast and robust way.

Author

Ammar Qammaz (AmmarkoV)

6.61.2 Enumeration Type Documentation

6.61.2.1 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

6.61.3 Function Documentation

6.61.3.1 int scanFor_audioFiles (const 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

6.62 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 (const char ∗str, unsigned int strLength)
  Scan a string for one of the words of the firstLines word set.

6.62.1 Function Documentation

6.62.1.1 int scanFor_firstLines (const 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

6.63 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 (const char ∗str, unsigned int strLength)

  Scan a string for one of the words of the firstLines word set.

6.63.1 Detailed Description

A tool that scans for a string in a very fast and robust way.

Author

Ammar Qammaz (AmmarkoV)

6.63.2 Enumeration Type Documentation

6.63.2.1 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

6.63.3 Function Documentation

6.63.3.1 int scanFor_firstLines (const 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

6.64 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 (const char ∗str, unsigned int strLength)

  Scan a string for one of the words of the httpHeader word set.

6.64.1 Function Documentation

6.64.1.1 int scanFor_httpHeader (const 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

6.65 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 (const char ∗str, unsigned int strLength)

  Scan a string for one of the words of the httpHeader word set.

6.65.1 Detailed Description

A tool that scans for a string in a very fast and robust way.

Author

Ammar Qammaz (AmmarkoV)

6.65.2 Enumeration Type Documentation

6.65.2.1 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

6.65.3 Function Documentation

6.65.3.1 int scanFor_httpHeader (const 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

6.66 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 (const char ∗str, unsigned int strLength)

  Scan a string for one of the words of the imageFiles word set.

6.66.1 Function Documentation

6.66.1.1 int scanFor_imageFiles (const 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

6.67 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 (const char ∗str, unsigned int strLength)

  Scan a string for one of the words of the imageFiles word set.

6.67.1 Detailed Description

A tool that scans for a string in a very fast and robust way.

Author

Ammar Qammaz (AmmarkoV)

6.67.2 Enumeration Type Documentation

6.67.2.1 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

6.67.3 Function Documentation

6.67.3.1 int scanFor_imageFiles (const 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

6.68 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 (const char ∗str, unsigned int strLength)

  Scan a string for one of the words of the postHeader word set.

6.68.1 Function Documentation

6.68.1.1 int scanFor_postHeader (const 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

6.69 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 (const char ∗str, unsigned int strLength)

  Scan a string for one of the words of the postHeader word set.

6.69.1 Detailed Description

A tool that scans for a string in a very fast and robust way.

Author

Ammar Qammaz (AmmarkoV)

6.69.2 Enumeration Type Documentation

6.69.2.1 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

6.69.3 Function Documentation

6.69.3.1 int scanFor_postHeader (const 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

6.70 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 (const char ∗str, unsigned int strLength)

  Scan a string for one of the words of the textFiles word set.

6.70.1 Function Documentation

6.70.1.1 int scanFor_textFiles (const 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

6.71 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_HTM, 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 (const char ∗str, unsigned int strLength)

  Scan a string for one of the words of the textFiles word set.

6.71.1 Detailed Description

A tool that scans for a string in a very fast and robust way.

Author

Ammar Qammaz (AmmarkoV)

6.71.2 Enumeration Type Documentation

6.71.2.1 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

6.71.3 Function Documentation

6.71.3.1 int scanFor_textFiles (const 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

6.72 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 (const char ∗str, unsigned int strLength)

  Scan a string for one of the words of the videoFiles word set.

6.72.1 Function Documentation

6.72.1.1 int scanFor_videoFiles (const 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

6.73 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 (const char ∗str, unsigned int strLength)

  Scan a string for one of the words of the videoFiles word set.

6.73.1 Detailed Description

A tool that scans for a string in a very fast and robust way.

Author

Ammar Qammaz (AmmarkoV)

6.73.2 Enumeration Type Documentation

6.73.2.1 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

6.73.3 Function Documentation

6.73.3.1 int scanFor_videoFiles (const 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

6.74 src/AmmServerlib/threads/clientServer.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 "clientServer.h"
#include "threadedServer.h"
#include "../tools/directory_lists.h"
#include "../network/file_server.h"
#include "../network/sendHTTPHeader.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 clientServer.c:

Functions

- 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.

6.74.1 Function Documentation

6.74.1.1 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-Thread	with information to pass to the new thread (prespawned or not)

Return values

This	function returns 0

START OF CLIENT IS NOT ON IP-BANNED-LIST!

Here is the call graph for this function:

6.74.1.2 int ServeClientKeepAliveLoop (struct AmmServer_Instance ∗ *instance,* struct HTTPTransaction ∗ *transaction*) [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:

6.75 src/AmmServerlib/threads/clientServer.h File Reference

This is the entry point to serve a client that picks a prespawned thread or creates a fresh new one and then handles the requests..

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.

6.75.1 Detailed Description

This is the entry point to serve a client that picks a prespawned thread or creates a fresh new one and then handles the requests..

Author

Ammar Qammaz (AmmarkoV)

6.75.2 Function Documentation

6.75.2.1 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-Thread	with information to pass to the new thread (prespawned or not)

Return values

This	function returns 0

START OF CLIENT IS NOT ON IP-BANNED-LIST!

Here is the call graph for this function:

6.76 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 "../server_configuration.h"
#include "../threads/clientServer.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)

  Create a new Thread that will serve the incoming client socket connection.

6.76.1 Macro Definition Documentation

6.76.1.1 #define MAX_TRIES_TO_FIND_A_THREAD_ID 5

6.76.1.2 #define WEIRD_THING_THAT_WORKS 1

6.76.2 Function Documentation

6.76.2.1 unsigned int FindAProperThreadID (struct AmmServer_Instance ∗ *instance,* int ∗ *success*)

Here is the call graph for this function:

6.76.2.2 int SpawnThreadToServeNewClient (struct AmmServer_Instance ∗ *instance,* int *clientsock,* struct sockaddr_in *client,* unsigned int *clientlen*)

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	

Bug There might be issues with the way the compiler optimizes the code that waits for the stack to be read before continuing on from the main thread..

Here is the call graph for this function:

6.77 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)

  Create a new Thread that will serve the incoming client socket connection.

6.77.1 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)

6.77.2 Function Documentation

6.77.2.1 int SpawnThreadToServeNewClient (struct AmmServer_Instance ∗ *instance,* int *clientsock,* struct sockaddr_in *client,* unsigned int *clientlen*)

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	

Bug There might be issues with the way the compiler optimizes the code that waits for the stack to be read before continuing on from the main thread..

Here is the call graph for this function:

6.78 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 "../tools/logs.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.

6.78.1 Function Documentation

6.78.1.1 void∗ PreSpawnedThread (void ∗ *ptr*)

Here is the call graph for this function:

6.78.1.2 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	

Here is the call graph for this function:

6.78.1.3 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	

Here is the call graph for this function:

6.79 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.

6.79.1 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 ?

6.79.2 Function Documentation

6.79.2.1 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	

Here is the call graph for this function:

6.79.2.2 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	

Here is the call graph for this function:

6.80 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 "../network/sendHTTPHeader.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.
- void ∗ MainHTTPServerThread (void ∗ptr)
- int StartHTTPServer (struct AmmServer_Instance ∗instance, const char ∗ip, unsigned int port, const char ∗root_path, const char ∗templates_path)

  Start HTTP server.
- int StopHTTPServer (struct AmmServer_Instance ∗instance)

  Stop a running HTTP server , unbind ports , deallocate structures etc.

6.80.1 Function Documentation

6.80.1.1 int HTTPServerIsRunning (struct AmmServer_Instance ∗ *instance*)

Ask if the HTTP server is running.

Parameters

An	AmmarServer Instance

Return values

1=Success,0=Failure	

6.80.1.2 void∗ MainHTTPServerThread (void ∗ *ptr*)

Here is the call graph for this function:

6.80.1.3 int StartHTTPServer (struct AmmServer_Instance ∗ *instance,* const char ∗ *ip,* unsigned int *port,* const char ∗ *root_path,* const 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:

6.80.1.4 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:

6.81 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

- int StartHTTPServer (struct AmmServer_Instance ∗instance, const char ∗ip, unsigned int port, const char ∗root_path, const 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.

6.81.1 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)

6.81.2 Function Documentation

6.81.2.1 int HTTPServerIsRunning (struct AmmServer_Instance ∗ *instance*)

Ask if the HTTP server is running.

Parameters

An	AmmarServer Instance

Return values

1=Success,0=Failure	

6.81.2.2 int StartHTTPServer (struct **AmmServer_Instance** ∗ *instance,* const char ∗ *ip,* unsigned int *port,* const char ∗ *root_path,* const 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:

6.81.2.3 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:

6.82 src/AmmServerlib/threads/threadInitHelper.c File Reference

```
#include "threadInitHelper.h"
#include "../tools/logs.h"
#include <stdio.h>
#include <unistd.h>
#include "../server_configuration.h"
```

Include dependency graph for threadInitHelper.c:

Macros

- #define SLEEP_FOR_N_NANOSECONDS_WAITING_STACK_MESSAGE 10

6.82.1 Macro Definition Documentation

6.82.1.1 #define SLEEP_FOR_N_NANOSECONDS_WAITING_STACK_MESSAGE 10

6.83 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:

6.83.1 Detailed Description

Helper Functions to help with passing messages around ..

Author

Ammar Qammaz (AmmarkoV)

6.84 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 "logs.h"
#include "directory_lists.h"
#include "http_tools.h"
#include <sys/stat.h>
#include <errno.h>
#include "../../../public_html/templates/directoryListStart.html"
#include "../../../public_html/templates/directoryListEnd.html"
```

Include dependency graph for directory_lists.c:

Macros

- #define tag_pre_image "<tr><td><img src=\"/"
- #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.

Variables

- char ∗ starting =
- char ∗ ending =

6.84.1 Macro Definition Documentation

6.84.1.1 #define tag_after_image "\">"

6.84.1.2 #define tag_pre_image "<tr><td><img src=\"/"

6.84.2 Function Documentation

6.84.2.1 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 , needs a lot of work

Here is the call graph for this function:

6.84.2.2 char∗ path_cat (const char ∗ *str1,* char ∗ *str2*)

6.84.3 Variable Documentation

6.84.3.1 char∗ ending =

6.84.3.2 char∗ starting =

6.85 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.

6.85.1 Detailed Description

Basic file server functionality of AmmarServer.

Author

Ammar Qammaz (AmmarkoV)

6.85.2 Function Documentation

6.85.2.1 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 , needs a lot of work

Here is the call graph for this function:

6.86 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/archiveFiles.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 (const char ∗filename)

    Check if file Exists.
- char DirectoryExistsAmmServ (const char ∗dirpath)

    Check if directory Exists.
- int GetContentTypeForExtension (const char ∗theextension, char ∗content_type, unsigned int contentTypeLength)
- int GetExtentionType (const char ∗theextension)

    Convert an Extension Type to a contentTypeEnumerator.
- void convertToUpperCase (char ∗sPtr)
- int GetContentType (const 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 CheckIfFileIsVideo (const char ∗filename)
- 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 (const char ∗input, const 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, unsigned int indexFileLength)
- 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.

6.86.1 Function Documentation

6.86.1.1 int CheckHTTPHeaderCategory (char ∗ *line,* unsigned int *line_length,* char ∗ *potential_strCAPS,* unsigned int ∗ *payload_start*)

Here is the call graph for this function:

6.86.1.2 int CheckHTTPHeaderCategoryAllCaps (char ∗ *lineCAPS,* unsigned int *line_length,* char ∗ *potential_strCAPS,* unsigned int ∗ *payload_start*)

Here is the call graph for this function:

6.86.1.3 int CheckIfFileIsVideo (const char ∗ *filename*)

Here is the call graph for this function:

6.86.1.4 void convertToUpperCase (char ∗ *sPtr*)

6.86.1.5 char DirectoryExistsAmmServ (const char ∗ *dirpath*)

Check if directory Exists.

Parameters

Path	to directory

Return values

1=Exists,0=Does	not Exist

6.86.1.6 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	string length
Output	string
Input	Maximum output length

Return values

1=Success,0=Failure	

6.86.1.7 char FileExistsAmmServ (const char ∗ *filename*)

Check if file Exists.

Parameters

Path	to file

Return values

1=Exists,0=Does	not Exist

6.86.1.8 int FilenameStripperOk (char ∗ *filename*)

Strip filename and security check it.

Parameters

Pointer	to string pointer to be analyzed

Return values

1=Ok,0=Failed	

6.86.1.9 int FindIndexFile (struct AmmServer_Instance ∗ *instance,* char ∗ *webserver_root,* char ∗ *directory,* char ∗ *indexfile,* unsigned int *indexFileLength*)

Here is the call graph for this function:

6.86.1.10 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

ClientID	or ,0=Failure

Here is the call graph for this function:

6.86.1.11 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	

6.86.1.12 int GetContentType (const 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:

6.86.1.13 int GetContentTypeForExtension (const char ∗ *theextension,* char ∗ *content_type,* unsigned int *contentTypeLength*)

Here is the call graph for this function:

6.86.1.14 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:

6.86.1.15 int GetExtentionType (const char ∗ *theextension*)

Convert an Extension Type to a contentTypeEnumerator.

Parameters

String	with the extension type

Return values

contentTypeEnumerator	

6.86.1.16 unsigned int GetIntFromHTTPHeaderFieldPayload (char ∗ *request,* unsigned int *request_length*)

Here is the call graph for this function:

6.86.1.17 char∗ GetNewStringFromHTTPHeaderFieldPayload (char ∗ *request,* unsigned int *request_length*)

Here is the call graph for this function:

6.86.1.18 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	

6.86.1.19 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

Bug : Check for success or failure on RequestHTTPWebPage and return an appropriate return value

Here is the call graph for this function:

6.86.1.20 int seek_blank_char (char ∗ *input,* char ∗ *input_end*)

6.86.1.21 int seek_non_blank_char (char ∗ *input,* char ∗ *input_end*)

6.86.1.22 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:

6.86.1.23 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:

6.86.1.24 int StripGETRequestQueryAndFragment (char ∗ *filename,* char ∗ *query,* unsigned int *max_query_length*)

6.86.1.25 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	

6.86.1.26 int StripVariableFromGETorPOSTString (const char ∗ *input,* const char ∗ *var_id,* char ∗ *var_val,* unsigned int *var_val_length*)

Bug StripVariableFromGETorPOSTString does not have a high quality implementation

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:

6.86.1.27 int stristr (char ∗ *str1CAPS,* unsigned int *str1_length,* char ∗ *str2CAPS,* unsigned int *str2_length,* unsigned int ∗ *pos_found*) [inline]

6.86.1.28 int stristr2Caps (char ∗ *str1,* unsigned int *str1_length,* char ∗ *str2CAPS,* unsigned int *str2_length,* unsigned int ∗ *pos_found*) [inline]

6.86.1.29 int strToUpcase (char ∗ *strTarget,* char ∗ *strSource,* unsigned int *strLength*)

6.86.1.30 int trim_last_empty_chars (char ∗ *input,* unsigned int *input_length*)

6.87 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 (const char ∗filename)

  Check if file Exists.
- char DirectoryExistsAmmServ (const char ∗dirpath)

  Check if directory Exists.
- int GetExtentionType (const char ∗theextension)

  Convert an Extension Type to a contentTypeEnumerator.
- int GetContentType (const 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 CheckIfFileIsVideo (const char ∗filename)
- int FindIndexFile (struct AmmServer_Instance ∗instance, char ∗webserver_root, char ∗directory, char ∗indexfile, unsigned int indexFileLength)
- int StripGETRequestQueryAndFragment (char ∗filename, char ∗query, unsigned int max_query_length)
- int StripVariableFromGETorPOSTString (const char ∗input, const char ∗var_id, char ∗var_val, unsigned int var_val_length)

- 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)

  Filenames may contain ///// with an arbitrary number of slashes , 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.

6.87.1 Detailed Description

A collection of tools required by the server and gathered here since they do a very specific job.

Author

Ammar Qammaz (AmmarkoV)

6.87.2 Typedef Documentation

6.87.2.1 typedef unsigned int **contentType**

6.87.3 Enumeration Type Documentation

6.87.3.1 enum **contentTypeEnumerator**

Enumerator

NO_FILETYPE

RESERVED_CTE_VALUE

TEXT

IMAGE

AUDIO

VIDEO

EXECUTABLE

FOLDER

6.87.4 Function Documentation

6.87.4.1 int CheckHTTPHeaderCategory (char ∗ *line,* unsigned int *line_length,* char ∗ *potential_strCAPS,* unsigned int ∗ *payload_start*)

Here is the call graph for this function:

6.87.4.2 int CheckHTTPHeaderCategoryAllCaps (char ∗ *lineCAPS,* unsigned int *line_length,* char ∗ *potential_strCAPS,* unsigned int ∗ *payload_start*)

Here is the call graph for this function:

6.87.4.3 int CheckIfFileIsVideo (const char ∗ *filename*)

Here is the call graph for this function:

6.87.4.4 char DirectoryExistsAmmServ (const char ∗ *dirpath*)

Check if directory Exists.

Parameters

Path	to directory

Return values

1=Exists,0=Does	not Exist

6.87.4.5 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	string length
Output	string
Input	Maximum output length

Return values

1=Success,0=Failure	

6.87.4.6 char FileExistsAmmServ (const char ∗ *filename*)

Check if file Exists.

Parameters

Path	to file

Return values

1=Exists,0=Does	not Exist

6.87.4.7 int FilenameStripperOk (char ∗ *filename*)

Strip filename and security check it.

Parameters

Pointer	to string pointer to be analyzed

Return values

1=Ok,0=Failed	

6.87.4.8 int FindIndexFile (struct AmmServer_Instance ∗ *instance,* char ∗ *webserver_root,* char ∗ *directory,* char ∗ *indexfile,* unsigned int *indexFileLength*)

Here is the call graph for this function:

6.87.4.9 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

ClientID	or ,0=Failure

Here is the call graph for this function:

6.87.4.10 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	

6.87.4.11 int GetContentType (const 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:

6.87.4.12 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:

6.87.4.13 int GetExtentionType (const char ∗ *theextension*)

Convert an Extension Type to a contentTypeEnumerator.

Parameters

String	with the extension type

Return values

contentTypeEnumerator	

6.87.4.14 unsigned int GetIntFromHTTPHeaderFieldPayload (char ∗ *request,* unsigned int *request_length*)

Here is the call graph for this function:

6.87.4.15 char∗ GetNewStringFromHTTPHeaderFieldPayload (char ∗ *request,* unsigned int *request_length*)

Here is the call graph for this function:

6.87.4.16 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	

6.87.4.17 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

Bug : Check for success or failure on RequestHTTPWebPage and return an appropriate return value

Here is the call graph for this function:

6.87.4.18 int seek_blank_char (char ∗ *input,* char ∗ *input_end*)

6.87.4.19 int seek_non_blank_char (char ∗ *input,* char ∗ *input_end*)

6.87.4.20 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:

6.87.4.21 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:

6.87.4.22 int StripGETRequestQueryAndFragment (char ∗ *filename,* char ∗ *query,* unsigned int *max_query_length*)

6.87.4.23 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	

6.87.4.24 int StripVariableFromGETorPOSTString (const char ∗ *input,* const char ∗ *var_id,* char ∗ *var_val,* unsigned int *var_val_length*)

Bug StripVariableFromGETorPOSTString does not have a high quality implementation

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:

6.87.4.25 int strToUpcase (char ∗ *strTarget,* char ∗ *strSource,* unsigned int *strLength*)

6.87.4.26 int trim_last_empty_chars (char ∗ *input,* unsigned int *input_length*)

6.88 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)

6.88.1 Function Documentation

6.88.1.1 int AccessLogAppend (char ∗ *IP,* char ∗ *DateStr,* char ∗ *Request,* unsigned int *ResponseCode,* unsigned long *ResponseLength,* char ∗ *Location,* char ∗ *Useragent*)

6.88.1.2 void error (char ∗ *msg*)

Log Function to output Errors.

Parameters

String	with message To log

6.88.1.3 int ErrorLogAppend (char ∗ *IP,* char ∗ *DateStr,* char ∗ *Request,* unsigned int *ResponseCode,* unsigned long *ResponseLength,* char ∗ *Location,* char ∗ *Useragent*)

6.88.1.4 void warning (char ∗ *msg*)

Log Function to output warnings.

Parameters

String	with message To log

6.89 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 ∗/
- #define logEcho() fprintf(stderr," Reached %s , %u \n ", __FILE__, __LINE__);

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)

6.89.1 Detailed Description

Logging functions.

Author

Ammar Qammaz (AmmarkoV)

6.89.2 Macro Definition Documentation

6.89.2.1 #define BLACK "\033[30m" /∗ Black ∗/

6.89.2.2 #define BLUE "\033[34m" /∗ Blue ∗/

6.89.2.3 #define BOLDBLACK "\033[1m\033[30m" /∗ Bold Black ∗/

6.89.2.4 #define BOLDBLUE "\033[1m\033[34m" /∗ Bold Blue ∗/

6.89.2.5 #define BOLDCYAN "\033[1m\033[36m" /∗ Bold Cyan ∗/

6.89.2.6 #define BOLDGREEN "\033[1m\033[32m" /∗ Bold Green ∗/

6.89.2.7 #define BOLDMAGENTA "\033[1m\033[35m" /∗ Bold Magenta ∗/

6.89.2.8 #define BOLDRED "\033[1m\033[31m" /∗ Bold Red ∗/

6.89.2.9 #define BOLDWHITE "\033[1m\033[37m" /∗ Bold White ∗/

6.89.2.10 #define BOLDYELLOW "\033[1m\033[33m" /∗ Bold Yellow ∗/

6.89.2.11 #define CYAN "\033[36m" /∗ Cyan ∗/

6.89.2.12 #define GREEN "\033[32m" /∗ Green ∗/

6.89.2.13 #define logEcho( ) fprintf(stderr," Reached %s , %u \n ", __FILE__, __LINE__);

6.89.2.14 #define MAGENTA "\033[35m" /∗ Magenta ∗/

6.89.2.15 #define NORMAL "\033[0m"

6.89.2.16 #define RED "\033[31m" /∗ Red ∗/

6.89.2.17 #define WHITE "\033[37m" /∗ White ∗/

6.89.2.18 #define YELLOW "\033[33m" /∗ Yellow ∗/

6.89.3 Function Documentation

6.89.3.1 int AccessLogAppend (char ∗ *IP,* char ∗ *DateStr,* char ∗ *Request,* unsigned int *ResponseCode,* unsigned long *ResponseLength,* char ∗ *Location,* char ∗ *Useragent*)

6.89.3.2 void error (char ∗ *msg*)

Log Function to output Errors.

Parameters

String	with message To log

6.89.3.3 int ErrorLogAppend (char ∗ *IP,* char ∗ *DateStr,* char ∗ *Request,* unsigned int *ResponseCode,* unsigned long *ResponseLength,* char ∗ *Location,* char ∗ *Useragent*)

6.89.3.4 void warning (char ∗ *msg*)

Log Function to output warnings.

Parameters

String	with message To log

6.90 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, unsigned int maxOutput, 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"}
- const char ∗ months [] = {"Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec"}

6.90.1 Function Documentation

6.90.1.1 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..

6.90.1.2 int GetDateString (char ∗ *output,* unsigned int *maxOutput,* 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	

6.90.1.3 unsigned long GetTickCountAmmServ ( )

GetTickCount like call for functions wanting to get monotonic values in milliseconds.

Return values

Milliseconds	

6.90.1.4 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	

6.90.2 Variable Documentation

6.90.2.1 const char∗ days[ ] = {"Sun","Mon","Tue","Wed","Thu","Fri","Sat"}

6.90.2.2 const char∗ months[ ] = {"Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec"}

6.91 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: 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, unsigned int maxOutput, 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.

6.91.1 Detailed Description

Timer functions.

Author

Ammar Qammaz (AmmarkoV)

6.91.2 Function Documentation

6.91.2.1 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..

6.91.2.2 int GetDateString (char ∗ *output,* unsigned int *maxOutput,* 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	

6.91.2.3 unsigned long GetTickCountAmmServ ( )

GetTickCount like call for functions wanting to get monotonic values in milliseconds.

Return values

Milliseconds	

6.91.2.4 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	

6.92 src/AmmServerlib/version.h File Reference

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

Macros

- #define RC_FILEVERSION 0,29,299,1497
- #define RC_FILEVERSION_STRING "0, 29, 299, 1497\0"

6.92.1 Macro Definition Documentation

6.92.1.1 #define RC_FILEVERSION 0,29,299,1497

6.92.1.2 #define RC_FILEVERSION_STRING "0, 29, 299, 1497\0"

6.93 src/Services/HabChan/board.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <unistd.h>
#include "state.h"
#include "thread.h"
#include "../../AmmServerlib/AmmServerlib.h"
#include "../../AmmServerlib/InputParser/InputParser_C.h"
```

Include dependency graph for board.c:

Functions

- void ∗ prepareBoardIndexView (struct AmmServer_DynamicRequest ∗rqst)
- int loadBoardSettings (char ∗boardName, struct board ∗ourBoard)
- int addBoardToSite (struct site ∗targetSite, char ∗boardName)

6.93.1 Function Documentation

6.93.1.1 int addBoardToSite (struct site ∗ *targetSite,* char ∗ *boardName*)

Here is the call graph for this function:

6.93.1.2 int loadBoardSettings (char ∗ *boardName,* struct board ∗ *ourBoard*)

Here is the call graph for this function:

6.93.1.3 void∗ prepareBoardIndexView (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.94 src/Services/HabChan/board.h File Reference

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

Include dependency graph for board.h: This graph shows which files directly or indirectly include this file:

Functions

- void ∗ prepareBoardIndexView (struct AmmServer_DynamicRequest ∗rqst)
- int addBoardToSite (struct site ∗targetSite, char ∗boardName)

6.94.1 Function Documentation

6.94.1.1 int addBoardToSite (struct site ∗ *targetSite,* char ∗ *boardName*)

Here is the call graph for this function:

6.94.1.2 void∗ prepareBoardIndexView (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.95 src/Services/HabChan/main.h File Reference

6.96 src/Services/HabChan/postReceiver.c File Reference

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

Include dependency graph for postReceiver.c:

Functions

- void ∗ processPostReceiver (struct AmmServer_DynamicRequest ∗rqst)

6.96.1 Function Documentation

6.96.1.1 void∗ processPostReceiver (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.97 src/Services/HabChan/postReceiver.h File Reference

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

Include dependency graph for postReceiver.h: This graph shows which files directly or indirectly include this file:

Functions

- void ∗ processPostReceiver (struct AmmServer_DynamicRequest ∗rqst)

6.97.1 Function Documentation

6.97.1.1 void∗ processPostReceiver (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.98 src/Services/HabChan/state.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "../../AmmServerlib/AmmServerlib.h"
#include "../../AmmServerlib/hashmap/hashmap.h"
#include "../../AmmServerlib/InputParser/InputParser_C.h"
#include "state.h"
#include "board.h"
```

Include dependency graph for state.c:

Functions

- void ∗ debug_get_callback (struct AmmServer_DynamicRequest ∗rqst)
- int loadSite (char ∗filename)
- int unloadSite ()
- int addPostToThread (char ∗boardName, struct thread ∗newThread, struct post ∗newPost)

Variables

- struct AmmServer_Instance ∗ default_server =0
- struct AmmServer_Instance ∗ admin_server =0
- struct AmmServer_RequestOverride_Context GET_override ={{0}}
- struct hashMap ∗ boardHashMap =0
- struct hashMap ∗ threadHashMap =0
- struct site ourSite ={0}
- unsigned int threadIndexPageLength = 0
- char ∗ threadIndexPage = 0
- unsigned int threadIndexStartPageLength = 0
- char ∗ threadIndexStartPage = 0
- unsigned int threadIndexEndPageLength = 0
- char ∗ threadIndexEndPage = 0

6.98.1 Function Documentation

6.98.1.1 int addPostToThread (char ∗ *boardName,* struct thread ∗ *newThread,* struct post ∗ *newPost*)

6.98.1.2 void∗ debug_get_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

6.98.1.3 int loadSite (char ∗ *filename*)

Here is the call graph for this function:

6.98.1.4 int unloadSite ( )

Here is the call graph for this function:

6.98.2 Variable Documentation

6.98.2.1 struct AmmServer_Instance∗ admin_server =0

6.98.2.2 struct hashMap∗ boardHashMap =0

6.98.2.3 struct AmmServer_Instance∗ default_server =0

Dynamic content code ..! START!

6.98.2.4 struct AmmServer_RequestOverride_Context GET_override ={{0}}

6.98.2.5 struct site ourSite ={0}

6.98.2.6 struct hashMap∗ threadHashMap =0

6.98.2.7 char∗ threadIndexEndPage = 0

6.98.2.8 unsigned int threadIndexEndPageLength = 0

6.98.2.9 char∗ threadIndexPage = 0

6.98.2.10 unsigned int threadIndexPageLength = 0

6.98.2.11 char∗ threadIndexStartPage = 0

6.98.2.12 unsigned int threadIndexStartPageLength = 0

6.99 src/Services/HabChan/state.h File Reference

```
#include "../../AmmServerlib/AmmServerlib.h"
#include "../../AmmServerlib/hashmap/hashmap.h"
#include "state.h"
```

Include dependency graph for state.h: This graph shows which files directly or indirectly include this file:

Data Structures

- struct timestamp

  Timestamp for a cache item entry.
- struct post
- struct thread
- struct board
- struct site

Macros

- #define MAX_BOARDS 1000
- #define MAX_THREADS_PER_BOARD 1000
- #define LINE_MAX_LENGTH 1024
- #define MAX_STRING_SIZE 512

Enumerations

- enum FILETYPES_ENUM {
  FILETYPE_FORBIDDEN =0, FILETYPE_IMAGE, FILETYPE_AUDIO, FILETYPE_VIDEO_FILE,
  FILETYPE_VIDEO_YOUTUBE, NUMBER_OF_FILETYPES }

Functions

- int loadSite (char ∗filename)
- int unloadSite ()
- int addPostToThread (char ∗boardName, struct thread ∗newThread, struct post ∗newPost)

Variables

- struct AmmServer_Instance ∗ default_server
- struct AmmServer_Instance ∗ admin_server
- struct
  AmmServer_RequestOverride_Context GET_override
- struct site ourSite
- unsigned int threadIndexPageLength
- char ∗ threadIndexPage
- unsigned int threadIndexStartPageLength
- char ∗ threadIndexStartPage
- unsigned int threadIndexEndPageLength
- char ∗ threadIndexEndPage
- struct hashMap ∗ boardHashMap
- struct hashMap ∗ threadHashMap

6.99.1 Macro Definition Documentation

6.99.1.1 #define LINE_MAX_LENGTH 1024

6.99.1.2 #define MAX_BOARDS 1000

6.99.1.3 #define MAX_STRING_SIZE 512

6.99.1.4 #define MAX_THREADS_PER_BOARD 1000

6.99.2 Enumeration Type Documentation

6.99.2.1 enum FILETYPES_ENUM

Enumerator

FILETYPE_FORBIDDEN

FILETYPE_IMAGE

FILETYPE_AUDIO

FILETYPE_VIDEO_FILE

FILETYPE_VIDEO_YOUTUBE

NUMBER_OF_FILETYPES

6.99.3 Function Documentation

6.99.3.1 int addPostToThread (char * *boardName,* struct thread * *newThread,* struct post * *newPost*)

6.99.3.2 int loadSite (char * *filename*)

Here is the call graph for this function:

6.99.3.3 int unloadSite ( )

Here is the call graph for this function:

6.99.4 Variable Documentation

6.99.4.1 struct AmmServer_Instance* admin_server

6.99.4.2 struct hashMap* boardHashMap

6.99.4.3 struct AmmServer_Instance* default_server

Dynamic content code ..! START!

6.99.4.4 struct AmmServer_RequestOverride_Context GET_override

6.99.4.5 struct site ourSite

6.99.4.6 struct hashMap* threadHashMap

6.99.4.7 char* threadIndexEndPage

6.99.4.8 unsigned int threadIndexEndPageLength

6.99.4.9 char* threadIndexPage

6.99.4.10 unsigned int threadIndexPageLength

6.99.4.11 char* threadIndexStartPage

6.99.4.12 unsigned int threadIndexStartPageLength

6.100 src/Services/HabChan/thread.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <unistd.h>
#include "state.h"
#include "thread.h"
#include "../../AmmServerlib/AmmServerlib.h"
#include "../../AmmServerlib/InputParser/InputParser_C.h"
```

Include dependency graph for thread.c:

Functions

- void ∗ prepareThreadView (struct AmmServer_DynamicRequest ∗rqst)
- char ∗ mallocHTMLListOfThreadsOfBoard (const char ∗boardName, unsigned int ∗htmlLength)
- void ∗ prepareThreadIndexView (struct AmmServer_DynamicRequest ∗rqst)
- int loadThread (const char ∗threadName, struct board ∗ourBoard, struct thread ∗ourThread)
- int addThreadToBoard (const char ∗boardName, const char ∗threadName)

6.100.1 Function Documentation

6.100.1.1 int addThreadToBoard (const char ∗ *boardName,* const char ∗ *threadName*)

Here is the call graph for this function:

6.100.1.2 int loadThread (const char ∗ *threadName,* struct **board** ∗ *ourBoard,* struct **thread** ∗ *ourThread*)

Here is the call graph for this function:

6.100.1.3 char∗ mallocHTMLListOfThreadsOfBoard (const char ∗ *boardName,* unsigned int ∗ *htmlLength*)

Here is the call graph for this function:

6.100.1.4 void∗ prepareThreadIndexView (struct **AmmServer_DynamicRequest** ∗ *rqst*)

Here is the call graph for this function:

6.100.1.5 void∗ prepareThreadView (struct **AmmServer_DynamicRequest** ∗ *rqst*)

6.101 src/Services/HabChan/thread.h File Reference

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

Include dependency graph for thread.h: This graph shows which files directly or indirectly include this file:

Functions

- void ∗ prepareThreadView (struct AmmServer_DynamicRequest ∗rqst)
- void ∗ prepareThreadIndexView (struct AmmServer_DynamicRequest ∗rqst)
- int addThreadToBoard (const char ∗boardName, const char ∗threadName)

6.101.1 Function Documentation

6.101.1.1 int addThreadToBoard (const char ∗ *boardName,* const char ∗ *threadName*)

Here is the call graph for this function:

6.101.1.2 void∗ prepareThreadIndexView (struct **AmmServer_DynamicRequest** ∗ *rqst*)

Here is the call graph for this function:

6.101.1.3 void∗ prepareThreadView (struct AmmServer_DynamicRequest ∗ *rqst*)

6.102 src/Services/MyBlog/database.c File Reference

```
#include "database.h"
#include <sqlite3.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

Include dependency graph for database.c:

Functions

- int SQL_error (struct SQLiteSession ∗sqlserver, int rc, const char ∗msg, unsigned int line)
- int SQL_init (struct SQLiteSession ∗sqlserver, const char ∗dbFilename)
- int SQL_close (struct SQLiteSession ∗sqlserver)
- int SQL_getVersion (struct SQLiteSession ∗sqlserver)
- int SQL_createInitialTables (struct SQLiteSession ∗sqlserver)
- int appendPosts (void ∗rqstV, int argc, char ∗∗argv, char ∗∗azColName)
- int loadPostsFromSQL (struct SQLiteSession ∗sqlserver, struct website ∗websiteContext)

Variables

- struct website myblog ={0}
- struct SQLiteSession sqlserver ={0}

6.102.1 Function Documentation

6.102.1.1 int appendPosts (void ∗ *rqstV,* int *argc,* char ∗∗ *argv,* char ∗∗ *azColName*)

6.102.1.2 int loadPostsFromSQL (struct SQLiteSession ∗ *sqlserver,* struct website ∗ *websiteContext*)

Here is the call graph for this function:

6.102.1.3 int SQL_close (struct SQLiteSession ∗ *sqlserver*)

6.102.1.4 int SQL_createInitialTables (struct SQLiteSession ∗ *sqlserver*)

Here is the call graph for this function:

6.102.1.5 int SQL_error (struct SQLiteSession ∗ *sqlserver,* int *rc,* const char ∗ *msg,* unsigned int *line*)

6.102.1.6 int SQL_getVersion (struct SQLiteSession ∗ *sqlserver*)

6.102.1.7 int SQL_init (struct SQLiteSession ∗ *sqlserver,* const char ∗ *dbFilename*)

6.102.2 Variable Documentation

6.102.2.1 struct website myblog ={0}

6.102.2.2 struct SQLiteSession sqlserver ={0}

6.103 src/Services/MyBlog/database.h File Reference

```
#include <sqlite3.h>
#include "../../AmmServerlib/AmmServerlib.h"
```

Include dependency graph for database.h: This graph shows which files directly or indirectly include this file:

Data Structures

- struct SQLiteSession
- struct htmlContent
- struct socialLinks
- struct linkLabelItem
- struct menuItemList
- struct linkItemList
- struct widgetItem
- struct widgetItemList
- struct tagItem
- struct tagItemList
- struct postItem
- struct postItemList
- struct website

Macros

- #define MAX_STR 512
- #define MAX_CONTENT 16000
- #define MAX_MENU_ITEMS 10
- #define MAX_WIDGET_ITEMS 10
- #define MAX_TAGS_PER_POST 10
- #define CONTENT_BUFFER 16500

Functions

- int SQL_init (struct SQLiteSession ∗sqlserver, const char ∗dbFilename)
- int SQL_close (struct SQLiteSession ∗sqlserver)
- int loadPostsFromSQL (struct SQLiteSession ∗sqlserver, struct website ∗websiteContext)
- int SQL_createInitialTables (struct SQLiteSession ∗sqlserver)

Variables

- struct website myblog
- struct SQLiteSession sqlserver

6.103.1 Macro Definition Documentation

6.103.1.1 #define CONTENT_BUFFER 16500

6.103.1.2 #define MAX_CONTENT 16000

6.103.1.3 #define MAX_MENU_ITEMS 10

6.103.1.4 #define MAX_STR 512

6.103.1.5 #define MAX_TAGS_PER_POST 10

6.103.1.6 #define MAX_WIDGET_ITEMS 10

6.103.2 Function Documentation

6.103.2.1 int loadPostsFromSQL (struct **SQLiteSession** ∗ *sqlserver,* struct **website** ∗ *websiteContext*)

Here is the call graph for this function:

6.103.2.2 int SQL_close (struct **SQLiteSession** ∗ *sqlserver*)

6.103.2.3 int SQL_createInitialTables (struct **SQLiteSession** ∗ *sqlserver*)

Here is the call graph for this function:

6.103.2.4 int SQL_init (struct **SQLiteSession** ∗ *sqlserver,* const char ∗ *dbFilename*)

6.103.3 Variable Documentation

6.103.3.1 struct **website** myblog

6.103.3.2 struct **SQLiteSession** sqlserver

6.104 src/Services/MyBlog/index.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "index.h"
#include "database.h"
```

Include dependency graph for index.c:

Functions

- unsigned char ∗ getLeftBlogRollHTML (struct website ∗configuration)
- unsigned char ∗ getRightBlogRollHTML (struct website ∗configuration)
- unsigned char ∗ getFooterLinksHTML (struct website ∗configuration)
- unsigned char ∗ getMenuListHTML (struct website ∗configuration)
- unsigned char ∗ getWidgetListHTML (struct website ∗configuration)
- unsigned char ∗ getPostListHTML (struct website ∗configuration)
- int strlimcpy (char ∗output, unsigned int outputLimit, const char ∗source)
- int loadPosts (struct website ∗configuration)
- int setupMyBlog (struct website ∗configuration)
- int destroy_index_prototype ()
- unsigned char ∗ prepare_index_prototype (char ∗filename, struct website ∗configuration)
- void ∗ prepare_index (struct AmmServer_DynamicRequest ∗rqst)

Variables

- struct AmmServer_MemoryHandler ∗ indexPage =0

6.104.1 Function Documentation

6.104.1.1 int destroy_index_prototype ( )

Here is the call graph for this function:

6.104.1.2 unsigned char∗ getFooterLinksHTML (struct **website** ∗ *configuration*)

6.104.1.3 unsigned char∗ getLeftBlogRollHTML (struct **website** ∗ *configuration*)

6.104.1.4 unsigned char∗ getMenuListHTML (struct **website** ∗ *configuration*)

6.104.1.5 unsigned char∗ getPostListHTML (struct **website** ∗ *configuration*)

6.104.1.6 unsigned char∗ getRightBlogRollHTML (struct **website** ∗ *configuration*)

6.104.1.7 unsigned char∗ getWidgetListHTML (struct **website** ∗ *configuration*)

6.104.1.8 int loadPosts (struct **website** ∗ *configuration*)

Here is the call graph for this function:

6.104.1.9 void∗ prepare_index (struct **AmmServer_DynamicRequest** ∗ *rqst*)

6.104.1.10 unsigned char∗ prepare_index_prototype (char ∗ *filename,* struct **website** ∗ *configuration*)

Here is the call graph for this function:

6.104.1.11 int setupMyBlog (struct **website** ∗ *configuration*)

Here is the call graph for this function:

6.104.1.12 int strlimcpy (char ∗ *output,* unsigned int *outputLimit,* const char ∗ *source*)

6.104.2 Variable Documentation

6.104.2.1 struct **AmmServer_MemoryHandler**∗ indexPage =0

6.105 src/Services/MyBlog/index.h File Reference

`#include "../../AmmServerlib/AmmServerlib.h"`

Include dependency graph for index.h: This graph shows which files directly or indirectly include this file:

Functions

- int destroy_index_prototype ()
- void ∗ prepare_index (struct AmmServer_DynamicRequest ∗rqst)

6.105.1 Function Documentation

6.105.1.1 int destroy_index_prototype ( )

Here is the call graph for this function:

6.105.1.2 void∗ prepare_index (struct AmmServer_DynamicRequest ∗ *rqst*)

6.106 src/Services/MyBlog/tools/myblogTool.c File Reference

```
#include <sqlite3.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "../../../AmmServerlib/AmmServerlib.h"
```

Include dependency graph for myblogTool.c:

Data Structures

- struct SQLiteSession

Functions

- int SQL_error (struct SQLiteSession ∗sqlserver, int rc, const char ∗msg, unsigned int line)
- int SQL_init (struct SQLiteSession ∗sqlserver, const char ∗dbFilename)
- int SQL_close (struct SQLiteSession ∗sqlserver)
- int SQL_getVersion (struct SQLiteSession ∗sqlserver)
- int SQL_appendpost (struct SQLiteSession ∗sqlserver, const char ∗title, char ∗author, const char ∗data, unsigned int dataSize)
- int main (int argc, char ∗argv[])

Variables

- struct SQLiteSession sqlserver ={0}

6.106.1 Function Documentation

6.106.1.1 int main (int *argc,* char ∗ *argv[]*)

Here is the call graph for this function:

6.106.1.2 int SQL_appendpost (struct SQLiteSession ∗ *sqlserver,* const char ∗ *title,* char ∗ *author,* const char ∗ *data,* unsigned int *dataSize*)

6.106.1.3 int SQL_close (struct SQLiteSession ∗ *sqlserver*)

6.106.1.4 int SQL_error (struct SQLiteSession ∗ *sqlserver,* int *rc,* const char ∗ *msg,* unsigned int *line*)

6.106.1.5 int SQL_getVersion (struct SQLiteSession ∗ *sqlserver*)

6.106.1.6 int SQL_init (struct SQLiteSession ∗ *sqlserver,* const char ∗ *dbFilename*)

6.106.2 Variable Documentation

6.106.2.1 struct **SQLiteSession** sqlserver ={0}

6.107 src/Services/MyRemoteDesktop/xwd-1.0.5/clientwin.c File Reference

```
#include <X11/Xatom.h>
#include <X11/Xlib.h>
#include "clientwin.h"
```

Include dependency graph for clientwin.c:

Functions

- Window Find_Client (Display ∗dpy, Window root, Window subwin)

6.107.1 Function Documentation

6.107.1.1 Window Find_Client (Display ∗ *dpy,* Window *root,* Window *subwin*)

6.108 src/Services/MyRemoteDesktop/xwd-1.0.5/clientwin.h File Reference

```
#include <X11/Xlib.h>
```

Include dependency graph for clientwin.h: This graph shows which files directly or indirectly include this file:

Functions

- Window Find_Client (Display ∗dpy, Window root, Window target_win)

6.108.1 Function Documentation

6.108.1.1 Window Find_Client (Display ∗ *dpy,* Window *root,* Window *target_win*)

6.109 src/Services/MyRemoteDesktop/xwd-1.0.5/config.h File Reference

Macros

- #define HAVE_INTTYPES_H 1
- #define HAVE_MEMORY_H 1
- #define HAVE_STDINT_H 1
- #define HAVE_STDLIB_H 1
- #define HAVE_STRINGS_H 1
- #define HAVE_STRING_H 1
- #define HAVE_SYS_STAT_H 1
- #define HAVE_SYS_TYPES_H 1
- #define HAVE_UNISTD_H 1
- #define PACKAGE "xwd"
- #define PACKAGE_BUGREPORT "https://bugs.freedesktop.org/enter_bug.cgi?product=xorg"
- #define PACKAGE_NAME "xwd"
- #define PACKAGE_STRING "xwd 1.0.5"
- #define PACKAGE_TARNAME "xwd"
- #define PACKAGE_URL ""
- #define PACKAGE_VERSION "1.0.5"
- #define PACKAGE_VERSION_MAJOR 1

- #define PACKAGE_VERSION_MINOR 0
- #define PACKAGE_VERSION_PATCHLEVEL 5
- #define STDC_HEADERS 1
- #define VERSION "1.0.5"

6.109.1 Macro Definition Documentation

6.109.1.1 #define HAVE_INTTYPES_H 1

6.109.1.2 #define HAVE_MEMORY_H 1

6.109.1.3 #define HAVE_STDINT_H 1

6.109.1.4 #define HAVE_STDLIB_H 1

6.109.1.5 #define HAVE_STRING_H 1

6.109.1.6 #define HAVE_STRINGS_H 1

6.109.1.7 #define HAVE_SYS_STAT_H 1

6.109.1.8 #define HAVE_SYS_TYPES_H 1

6.109.1.9 #define HAVE_UNISTD_H 1

6.109.1.10 #define PACKAGE "xwd"

6.109.1.11 #define PACKAGE_BUGREPORT "https://bugs.freedesktop.org/enter_bug.cgi?product=xorg"

6.109.1.12 #define PACKAGE_NAME "xwd"

6.109.1.13 #define PACKAGE_STRING "xwd 1.0.5"

6.109.1.14 #define PACKAGE_TARNAME "xwd"

6.109.1.15 #define PACKAGE_URL ""

6.109.1.16 #define PACKAGE_VERSION "1.0.5"

6.109.1.17 #define PACKAGE_VERSION_MAJOR 1

6.109.1.18 #define PACKAGE_VERSION_MINOR 0

6.109.1.19 #define PACKAGE_VERSION_PATCHLEVEL 5

6.109.1.20 #define STDC_HEADERS 1

6.109.1.21 #define VERSION "1.0.5"

6.110 src/Services/MyRemoteDesktop/xwd-1.0.5/dsimple.c File Reference

```
#include <X11/Xos.h>
#include <X11/Xlib.h>
#include <X11/Xutil.h>
#include <X11/cursorfont.h>
#include <stdio.h>
#include <stdlib.h>
#include <stdarg.h>
#include "clientwin.h"
#include "dsimple.h"
```

Include dependency graph for dsimple.c:

Macros

- #define ARGC (∗rargc)
- #define OPTION argv[0]
- #define NXTOPTP ++argv, --argc>0
- #define NXTOPT if (++argv, --argc==0) usage()
- #define COPYOPT nargv++[0]=OPTION, nargc++

Functions

- char ∗ Get_Display_Name (int ∗pargc, char ∗∗argv)
- Display ∗ Open_Display (const char ∗display_name)
- void Setup_Display_And_Screen (int ∗argc, char ∗∗argv)
- void Setup_Null_Display_And_Screen ()
- void Close_Display (void)
- Window Select_Window_Args (int ∗rargc, char ∗∗argv)
- Window getRootWindow ()
- Window Select_Window (Display ∗disp, int descend)
- Window Window_With_Name (Display ∗disp, Window top, const char ∗name)
- void outl (char ∗msg,...)
- void Fatal_Error (char ∗msg,...)

Variables

- char ∗ program_name = "unknown_program"
- Display ∗ dpy = NULL
- int screen = 0

6.110.1 Macro Definition Documentation

6.110.1.1 #define ARGC (∗rargc)

6.110.1.2 #define COPYOPT nargv++[0]=OPTION, nargc++

6.110.1.3 #define NXTOPT if (++argv, --argc==0) usage()

6.110.1.4 #define NXTOPTP ++argv, --argc>0

6.110.1.5 #define OPTION argv[0]

6.110.2 Function Documentation

6.110.2.1 void Close_Display (void)

6.110.2.2 void Fatal_Error (char ∗ *msg,* ...)

Here is the call graph for this function:

6.110.2.3 char∗ Get_Display_Name (int ∗ *pargc,* char ∗∗ *argv*)

Here is the call graph for this function:

6.110.2.4 Window getRootWindow ( )

6.110.2.5 Display∗ Open_Display (const char ∗ *display_name*)

6.110.2.6 void outl (char ∗ *msg,* ...)

6.110.2.7 Window Select_Window (Display ∗ *disp,* int *descend*)

Here is the call graph for this function:

6.110.2.8 Window Select_Window_Args (int ∗ *rargc,* char ∗∗ *argv*)

Here is the call graph for this function:

6.110.2.9 void Setup_Display_And_Screen (int ∗ *argc,* char ∗∗ *argv*)

Here is the call graph for this function:

6.110.2.10 void Setup_Null_Display_And_Screen ( )

Here is the call graph for this function:

6.110.2.11 Window Window_With_Name (Display ∗ *disp,* Window *top,* const char ∗ *name*)

Here is the call graph for this function:

6.110.3 Variable Documentation

6.110.3.1 Display∗ dpy = NULL

6.110.3.2 char∗ program_name = "unknown_program"

6.110.3.3 int screen = 0

6.111 src/Services/MyRemoteDesktop/xwd-1.0.5/dsimple.h File Reference

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

Macros

- #define MAX(a, b) (((a)>(b))?(a):(b))
- #define MIN(a, b) (((a)<(b))?(a):(b))
- #define INIT_NAME
- #define X_USAGE

Functions

- char * Get_Display_Name (int *, char **)
- Display * Open_Display (const char *)
- void Setup_Display_And_Screen (int *, char **)
- void Setup_Null_Display_And_Screen ()
- Window getRootWindow ()
- void Close_Display (void)
- Window Select_Window_Args (int *, char **)
- void usage (void) _X_NORETURN
- Window Select_Window (Display *, int)
- Window Window_With_Name (Display *, Window, const char *)
- void Fatal_Error (char *,...) _X_NORETURN
- void outl (char *,...)

Variables

- char * program_name
- Display * dpy
- int screen

6.111.1 Macro Definition Documentation

6.111.1.1 #define INIT_NAME

Value:

```
program_name=argv[0]        /* use this in main to setup
                                                   program_name */
```

6.111.1.2 #define MAX(*a*, *b*) (((a)>(b))?(a):(b))

6.111.1.3 #define MIN(*a*, *b*) (((a)<(b))?(a):(b))

6.111.1.4 #define X_USAGE

Value:

```
"[host:display]"              /* X arguments handled by
                         Get_Display_Name */
```

6.111.2 Function Documentation

6.111.2.1 void Close_Display (void)

6.111.2.2 void Fatal_Error (char *, ...)

Here is the call graph for this function:

6.111.2.3 char∗ Get_Display_Name (int ∗ , char ∗∗)

Here is the call graph for this function:

6.111.2.4 Window getRootWindow ( )

6.111.2.5 Display∗ Open_Display (const char ∗)

6.111.2.6 void outl (char ∗ , ...)

6.111.2.7 Window Select_Window (Display ∗ , int)

Here is the call graph for this function:

6.111.2.8 Window Select_Window_Args (int ∗ , char ∗∗)

Here is the call graph for this function:

6.111.2.9 void Setup_Display_And_Screen (int ∗ , char ∗∗)

Here is the call graph for this function:

6.111.2.10 void Setup_Null_Display_And_Screen ( )

Here is the call graph for this function:

6.111.2.11 void usage (void)

6.111.2.12 Window Window_With_Name (Display ∗ , Window , const char ∗)

Here is the call graph for this function:

6.111.3 Variable Documentation

6.111.3.1 Display∗ dpy

6.111.3.2 char∗ program_name

6.111.3.3 int screen

6.112 src/Services/MyRemoteDesktop/xwd-1.0.5/list.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include "list.h"
```

Include dependency graph for list.c:

Functions

- void zero_list (list_ptr lp)
- int add_to_list (list_ptr lp, void ∗item)

- list_ptr new_list (void)
- list_ptr dup_list_head (list_ptr lp, int start_at_curr)
- unsigned int list_length (list_ptr lp)
- void ∗ delete_from_list (list_ptr lp, void ∗item)
- void delete_list (list_ptr lp, int free_items)
- void delete_list_destroying (list_ptr lp, void destructor(void ∗item))
- void ∗ first_in_list (list_ptr lp)
- void ∗ next_in_list (list_ptr lp)
- int list_is_empty (list_ptr lp)

6.112.1 Function Documentation

6.112.1.1 int add_to_list (**list_ptr** *lp,* void ∗ *item*)

Adds item to the list pointed to by lp. Finds the end of the list, then mallocs a new list node onto the end of the list. The item pointer in the new node is set to "item" passed in, and the next pointer in the new node is set to NULL.

Returns 1 if successful, 0 if the malloc failed.

6.112.1.2 void∗ delete_from_list (**list_ptr** *lp,* void ∗ *item*)

Scans thru list, looking for a node whose ptr.item is equal to the "item" passed in. "Equal" here means the same address - no attempt is made to match equivalent values stored in different locations. If a match is found, that node is deleted from the list. Storage for the node is freed, but not for the item itself. Returns a pointer to the item, so the caller can free it if it

so desires. If a match is not found, returns NULL.

6.112.1.3 void delete_list (**list_ptr** *lp,* int *free_items*)

Deletes each node in the list *except the head*. This allows the deletion of lists where the head is not malloced or created with new_list(). If free_items is true, each item pointed to

from the node is freed, in addition to the node itself.

6.112.1.4 void delete_list_destroying (**list_ptr** *lp,* void *destructorvoid* ∗*item*)

6.112.1.5 **list_ptr** dup_list_head (**list_ptr** *lp,* int *start_at_curr*)

Creates a new list head, pointing to the same list as the one passed in. If start_at_curr is TRUE, the new list's first item is the "current" item (as set by calls to first/next_in_list()). If start_at_curr is FALSE, the first item in the new list is the same as the first item in the old list. In either case, the curr pointer in the new list is the same as in the old list.

Returns a pointer to the new list head.

Here is the call graph for this function:

6.112.1.6 void∗ first_in_list (**list_ptr** *lp*)

Returns a ptr to the first *item* (not list node) in the list. Sets the list head node's curr ptr to the first node in the list.

Returns NULL if the list is empty.

6.112.1.7 int list_is_empty (list_ptr *lp*)

6.112.1.8 unsigned int list_length (list_ptr *lp*)

Returns the number of items in the list.

6.112.1.9 list_ptr new_list (void)

Creates a new list and sets its pointers to NULL.

Returns a pointer to the new list.

6.112.1.10 void∗ next_in_list (list_ptr *lp*)

Returns a ptr to the next *item* (not list node) in the list. Sets the list head node's curr ptr to the next node in the list. first_in_list must have been called prior.

Returns NULL if no next item.

6.112.1.11 void zero_list (list_ptr *lp*)

This file contains routines for manipulating generic lists. Lists are implemented with a "harness". In other words, each node in the list consists of two pointers, one to the data item and one to the next node in the list. The head of the list is the same struct as each node, but the "item" ptr is used to point to the current member of the list (used by the first_in_list and next_in_list functions).

Sets the pointers of the specified list to NULL.

6.113 src/Services/MyRemoteDesktop/xwd-1.0.5/list.h File Reference

`#include <X11/Xfuncproto.h>`
Include dependency graph for list.h: This graph shows which files directly or indirectly include this file:

Data Structures

- struct _list_item

Macros

- #define LESS -1
- #define EQUAL 0
- #define GREATER 1
- #define DUP_WHOLE_LIST 0
- #define START_AT_CURR 1

Typedefs

- typedef struct _list_item list
- typedef struct _list_item list_item
- typedef struct _list_item ∗ list_ptr
- typedef void(∗ DESTRUCT_FUNC_PTR)(void ∗)

Functions

- void zero_list (list_ptr)
- int add_to_list (list_ptr, void ∗)
- list_ptr new_list (void)
- list_ptr dup_list_head (list_ptr, int)
- unsigned int list_length (list_ptr)
- void ∗ delete_from_list (list_ptr, void ∗)
- void delete_list (list_ptr, int)
- void delete_list_destroying (list_ptr, DESTRUCT_FUNC_PTR)
- void ∗ first_in_list (list_ptr)
- void ∗ next_in_list (list_ptr)
- int list_is_empty (list_ptr)

6.113.1 Macro Definition Documentation

6.113.1.1 #define DUP_WHOLE_LIST 0

6.113.1.2 #define EQUAL 0

6.113.1.3 #define GREATER 1

6.113.1.4 #define LESS -1

This file contains routines for manipulating generic lists. Lists are implemented with a "harness". In other words, each node in the list consists of two pointers, one to the data item and one to the next node in the list. The head of the list is the same struct as each node, but the "item" ptr is used to point to the current member of the list (used by the first_in_list and next_in_list functions).

Copyright 1994 Hewlett-Packard Co. Copyright 1996, 1998 The Open Group

Permission to use, copy, modify, distribute, and sell this software and its documentation for any purpose is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

6.113.1.5 #define START_AT_CURR 1

6.113.2 Typedef Documentation

6.113.2.1 typedef void(∗ DESTRUCT_FUNC_PTR)(void ∗)

6.113.2.2 typedef struct _list_item list

6.113.2.3 typedef struct _list_item list_item

6.113.2.4 typedef struct _list_item ∗ list_ptr

6.113.3 Function Documentation

6.113.3.1 int add_to_list (list_ptr *lp,* void ∗ *item*)

Adds item to the list pointed to by lp. Finds the end of the list, then mallocs a new list node onto the end of the list. The item pointer in the new node is set to "item" passed in, and the next pointer in the new node is set to NULL.

Returns 1 if successful, 0 if the malloc failed.

6.113.3.2 void∗ delete_from_list (list_ptr *lp,* void ∗ *item*)

Scans thru list, looking for a node whose ptr.item is equal to the "item" passed in. "Equal" here means the same address - no attempt is made to match equivalent values stored in different locations. If a match is found, that node is deleted from the list. Storage for the node is freed, but not for the item itself. Returns a pointer to the item, so the caller can free it if it

so desires. If a match is not found, returns NULL.

6.113.3.3 void delete_list (list_ptr *lp,* int *free_items*)

Deletes each node in the list *except the head*. This allows the deletion of lists where the head is not malloced or created with new_list(). If free_items is true, each item pointed to

from the node is freed, in addition to the node itself.

6.113.3.4 void delete_list_destroying (list_ptr , DESTRUCT_FUNC_PTR)

6.113.3.5 list_ptr dup_list_head (list_ptr *lp,* int *start_at_curr*)

Creates a new list head, pointing to the same list as the one passed in. If start_at_curr is TRUE, the new list's first item is the "current" item (as set by calls to first/next_in_list()). If start_at_curr is FALSE, the first item in the new list is the same as the first item in the old list. In either case, the curr pointer in the new list is the same as in the old list.

Returns a pointer to the new list head.

Here is the call graph for this function:

6.113.3.6 void∗ first_in_list (list_ptr *lp*)

Returns a ptr to the first *item* (not list node) in the list. Sets the list head node's curr ptr to the first node in the list.

Returns NULL if the list is empty.

6.113.3.7 int list_is_empty (list_ptr)

6.113.3.8 unsigned int list_length (list_ptr *lp*)

Returns the number of items in the list.

6.113.3.9 list_ptr new_list (void)

Creates a new list and sets its pointers to NULL.

Returns a pointer to the new list.

6.113.3.10 void∗ next_in_list (list_ptr *lp*)

Returns a ptr to the next *item* (not list node) in the list. Sets the list head node's curr ptr to the next node in the list. first_in_list must have been called prior.

Returns NULL if no next item.

6.113.3.11 void zero_list (list_ptr *lp*)

This file contains routines for manipulating generic lists. Lists are implemented with a "harness". In other words, each node in the list consists of two pointers, one to the data item and one to the next node in the list. The head of the list is the same struct as each node, but the "item" ptr is used to point to the current member of the list (used by the first_in_list and next_in_list functions).

Sets the pointers of the specified list to NULL.

6.114 src/Services/MyRemoteDesktop/xwd-1.0.5/multiVis.c File Reference

```
#include <stdlib.h>
#include <X11/Xlib.h>
#include <X11/Xutil.h>
#include <X11/X.h>
#include <X11/Intrinsic.h>
#include <stdio.h>
#include "list.h"
#include "wsutils.h"
#include "multiVis.h"
```

Include dependency graph for multiVis.c:

Data Structures

- struct myBox
- struct my_XRegion
- struct image_win_type
- struct image_region_type

Macros

- #define SAME_REGIONS(s1, s2)
- #define MIN(a, b) ((a) < (b) ? a : b)
- #define MAX(a, b) ((a) > (b) ? a : b)
- #define RED_SHIFT 16
- #define GREEN_SHIFT 8
- #define BLUE_SHIFT 0
- #define STATIC_GRAY 0x01
- #define GRAY_SCALE 0x02
- #define PSEUDO_COLOR 0x04
- #define TRUE_COLOR 0x10
- #define DIRECT_COLOR 0x11

Typedefs

- typedef struct myBox myBOX
- typedef struct myBox myBoxRec
- typedef struct myBox ∗ myBoxPtr
- typedef struct my_XRegion myREGION

Functions

- void initFakeVisual (Visual *Vis)
- int GetMultiVisualRegions (Display *disp, Window srcRootWinid, int x, int y, unsigned int width, unsigned int height, int *transparentOverlays, int *numVisuals, XVisualInfo **pVisuals, int *numOverlayVisuals, OverlayInfo **pOverlayVisuals, int *numImageVisuals, XVisualInfo ***pImageVisuals, list_ptr *vis_regions, list_ptr *vis_image_regions, int *allImage)
- XImage * ReadAreaToImage (Display *disp, Window srcRootWinid, int x, int y, unsigned int width, unsigned int height, int numVisuals, XVisualInfo *pVisuals, int numOverlayVisuals, OverlayInfo *pOverlayVisuals, int numImageVisuals, XVisualInfo **pImageVisuals, list_ptr vis_regions, list_ptr vis_image_regions, int format, int allImage)
- int GetXVisualInfo (Display *display, int screen, int *transparentOverlays, int *numVisuals, XVisualInfo **pVisuals, int *numOverlayVisuals, OverlayInfo **pOverlayVisuals, int *numImageVisuals, XVisualInfo ***pImageVisuals)
- void FreeXVisualInfo (XVisualInfo *pVisuals, OverlayInfo *pOverlayVisuals, XVisualInfo **pImageVisuals)

6.114.1 Macro Definition Documentation

6.114.1.1 #define BLUE_SHIFT 0

6.114.1.2 #define DIRECT_COLOR 0x11

6.114.1.3 #define GRAY_SCALE 0x02

6.114.1.4 #define GREEN_SHIFT 8

6.114.1.5 #define MAX(*a*, *b*) ((a) > (b) ? a : b)

6.114.1.6 #define MIN(*a*, *b*) ((a) < (b) ? a : b)

6.114.1.7 #define PSEUDO_COLOR 0x04

6.114.1.8 #define RED_SHIFT 16

6.114.1.9 #define SAME_REGIONS(*s1*, *s2*)

Value:

```
((s1)->vis == (s2)->vis && (s1)->cmap == (s2)->cmap &&       \
     (s1)->x_vis <= (s2)->x_vis &&                      \
     (s1)->y_vis <= (s2)->y_vis &&                      \
     (s1)->x_vis + (s1)->width  >= (s2)->x_vis + (s2)->width && \
     (s1)->y_vis + (s1)->height >= (s2)->y_vis + (s2)->height)
```

Returns TRUE if the two structs pointed to have the same "vis" & "cmap" fields and s2 lies completely within s1. s1 and s2 can

point to structs of image_win_type or image_region_type.

6.114.1.10 #define STATIC_GRAY 0x01

6.114.1.11 #define TRUE_COLOR 0x10

6.114.2 Typedef Documentation

6.114.2.1 typedef struct myBox myBOX

6.114.2.2 typedef struct **myBox** ∗ **myBoxPtr**

6.114.2.3 typedef struct **myBox** **myBoxRec**

6.114.2.4 typedef struct **my_XRegion** **myREGION**

6.114.3 Function Documentation

6.114.3.1 void FreeXVisualInfo (XVisualInfo ∗ *pVisuals,* **OverlayInfo** ∗ *pOverlayVisuals,* XVisualInfo ∗∗ *pImageVisuals*)

6.114.3.2 int GetMultiVisualRegions (Display ∗ *,* Window *,* int *,* int *,* unsigned *int,* unsigned *int,* int ∗ *,* int ∗ *,* XVisualInfo ∗∗ *,* int ∗ *,* **OverlayInfo** ∗∗ *,* int ∗ *,* XVisualInfo ∗∗∗ *,* **list_ptr** ∗ *,* **list_ptr** ∗ *,* int ∗)

This file contains routines for manipulating generic lists. Lists are implemented with a "harness". In other words, each node in the list consists of two pointers, one to the data item and one to the next node in the list. The head of the list is the same struct as each node, but the "item" ptr is used to point to the current member of the list (used by the first_in_list and next_in_list functions).

Copyright 1994 Hewlett-Packard Co. Copyright 1996, 1998 The Open Group

Permission to use, copy, modify, distribute, and sell this software and its documentation for any purpose is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE OPEN GROUP BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of The Open Group shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from The Open Group.

Here is the call graph for this function:

6.114.3.3 int GetXVisualInfo (Display ∗ *display,* int *screen,* int ∗ *transparentOverlays,* int ∗ *numVisuals,* XVisualInfo ∗∗ *pVisuals,* int ∗ *numOverlayVisuals,* **OverlayInfo** ∗∗ *pOverlayVisuals,* int ∗ *numImageVisuals,* XVisualInfo ∗∗∗ *pImageVisuals*)

6.114.3.4 void initFakeVisual (Visual ∗ *Vis*)

6.114.3.5 XImage∗ ReadAreaToImage (Display ∗ *disp,* Window *srcRootWinid,* int *x,* int *y,* unsigned int *width,* unsigned int *height,* int *numVisuals,* XVisualInfo ∗ *pVisuals,* int *numOverlayVisuals,* **OverlayInfo** ∗ *pOverlayVisuals,* int *numImageVisuals,* XVisualInfo ∗∗ *pImageVisuals,* **list_ptr** *vis_regions,* **list_ptr** *vis_image_regions,* int *format,* int *allImage*)

----

end transparency

Here is the call graph for this function:

6.115 src/Services/MyRemoteDesktop/xwd-1.0.5/multiVis.h File Reference

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

Functions

- int GetMultiVisualRegions (Display ∗, Window, int, int, unsigned int, unsigned int, int ∗, int ∗, XVisualInfo ∗∗, int ∗, OverlayInfo ∗∗, int ∗, XVisualInfo ∗∗∗, list_ptr ∗, list_ptr ∗, int ∗)
- XImage ∗ ReadAreaToImage (Display ∗, Window, int, int, unsigned int, unsigned int, int, XVisualInfo ∗, int, OverlayInfo ∗, int, XVisualInfo ∗∗, list_ptr, list_ptr, int, int)
- void initFakeVisual (Visual ∗)

6.115.1 Function Documentation

6.115.1.1 int GetMultiVisualRegions (Display ∗ , Window , int , int , unsigned *int,* unsigned *int,* int ∗ , int ∗ , XVisualInfo ∗∗ , int ∗ , OverlayInfo ∗∗ , int ∗ , XVisualInfo ∗∗∗ , list_ptr ∗ , list_ptr ∗ , int ∗)

This file contains routines for manipulating generic lists. Lists are implemented with a "harness". In other words, each node in the list consists of two pointers, one to the data item and one to the next node in the list. The head of the list is the same struct as each node, but the "item" ptr is used to point to the current member of the list (used by the first_in_list and next_in_list functions).

Here is the call graph for this function:

6.115.1.2 void initFakeVisual (Visual ∗)

6.115.1.3 XImage∗ ReadAreaToImage (Display ∗ *disp,* Window *srcRootWinid,* int *x,* int *y,* unsigned int *width,* unsigned int *height,* int *numVisuals,* XVisualInfo ∗ *pVisuals,* int *numOverlayVisuals,* OverlayInfo ∗ *pOverlayVisuals,* int *numImageVisuals,* XVisualInfo ∗∗ *pImageVisuals,* list_ptr *vis_regions,* list_ptr *vis_image_regions,* int *format,* int *allImage*)

---——

end transparency

Here is the call graph for this function:

6.116 src/Services/MyRemoteDesktop/xwd-1.0.5/wsutils.h File Reference

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

Data Structures

- struct OverlayVisualPropertyRec
- struct OverlayInfo

Macros

- #define None 0
- #define NOT_FLEXIBLE 0
- #define FLEXIBLE 1
- #define SB_CMAP_TYPE_NORMAL 1
- #define SB_CMAP_TYPE_MONOTONIC 2
- #define SB_CMAP_TYPE_FULL 4

Functions

- int GetXVisualInfo (Display ∗display, int screen, int ∗transparentOverlays, int ∗numVisuals, XVisualInfo ∗∗pVisuals, int ∗numOverlayVisuals, OverlayInfo ∗∗pOverlayVisuals, int ∗numImageVisuals, XVisualInfo ∗∗∗pImageVisuals)
- void FreeXVisualInfo (XVisualInfo ∗pVisuals, OverlayInfo ∗pOverlayVisuals, XVisualInfo ∗∗pImageVisuals)
- int FindImagePlanesVisual (Display ∗display, int screen, int numImageVisuals, XVisualInfo ∗∗pImageVisuals, int sbCmapHint, int depthHint, int depthFlexibility, Visual ∗∗pImageVisualToUse, int ∗depthObtained)
- int FindOverlayPlanesVisual (Display ∗display, int screen, int numOverlayVisuals, OverlayInfo ∗pOverlayVisuals, int depthHint, int depthFlexibility, int transparentBackground, Visual ∗∗pOverlayVisualToUse, int ∗depthObtained, int ∗transparentColor)
- int CreateImagePlanesWindow (Display ∗display, int screen, Window parentWindow, int windowX, int windowY, int windowWidth, int windowHeight, int windowDepth, Visual ∗pImageVisualToUse, int argc, char ∗argv[], char ∗windowName, char ∗iconName, Window ∗imageWindow, Colormap ∗imageColormap, int ∗mustFreeImageColormap)
- int CreateOverlayPlanesWindow (Display ∗display, int screen, Window parentWindow, int windowX, int windowY, int windowWidth, int windowHeight, int windowDepth, Visual ∗pOverlayVisualToUse, int argc, char ∗argv[], char ∗windowName, char ∗iconName, int transparentBackground, int ∗transparentColor, Window ∗overlayWindow, Colormap ∗overlayColormap, int ∗mustFreeOverlayColormap)

6.116.1 Macro Definition Documentation

6.116.1.1 #define FLEXIBLE 1

6.116.1.2 #define None 0

6.116.1.3 #define NOT_FLEXIBLE 0

6.116.1.4 #define SB_CMAP_TYPE_FULL 4

6.116.1.5 #define SB_CMAP_TYPE_MONOTONIC 2

6.116.1.6 #define SB_CMAP_TYPE_NORMAL 1

6.116.2 Function Documentation

6.116.2.1 int CreateImagePlanesWindow (Display ∗ *display,* int *screen,* Window *parentWindow,* int *windowX,* int *windowY,* int *windowWidth,* int *windowHeight,* int *windowDepth,* Visual ∗ *pImageVisualToUse,* int *argc,* char ∗ *argv[ ],* char ∗ *windowName,* char ∗ *iconName,* Window ∗ *imageWindow,* Colormap ∗ *imageColormap,* int ∗ *mustFreeImageColormap*)

6.116.2.2 int CreateOverlayPlanesWindow (Display ∗ *display,* int *screen,* Window *parentWindow,* int *windowX,* int *windowY,* int *windowWidth,* int *windowHeight,* int *windowDepth,* Visual ∗ *pOverlayVisualToUse,* int *argc,* char ∗ *argv[ ],* char ∗ *windowName,* char ∗ *iconName,* int *transparentBackground,* int ∗ *transparentColor,* Window ∗ *overlayWindow,* Colormap ∗ *overlayColormap,* int ∗ *mustFreeOverlayColormap*)

6.116.2.3 int FindImagePlanesVisual (Display ∗ *display,* int *screen,* int *numImageVisuals,* XVisualInfo ∗∗ *pImageVisuals,* int *sbCmapHint,* int *depthHint,* int *depthFlexibility,* Visual ∗∗ *pImageVisualToUse,* int ∗ *depthObtained*)

6.116.2.4 int FindOverlayPlanesVisual (Display ∗ *display,* int *screen,* int *numOverlayVisuals,* OverlayInfo ∗ *pOverlayVisuals,* int *depthHint,* int *depthFlexibility,* int *transparentBackground,* Visual ∗∗ *pOverlayVisualToUse,* int ∗ *depthObtained,* int ∗ *transparentColor*)

6.116.2.5 void FreeXVisualInfo (XVisualInfo ∗ *pVisuals,* OverlayInfo ∗ *pOverlayVisuals,* XVisualInfo ∗∗ *pImageVisuals*)

6.116.2.6 int GetXVisualInfo (Display ∗ *display,* int *screen,* int ∗ *transparentOverlays,* int ∗ *numVisuals,* XVisualInfo ∗∗ *pVisuals,* int ∗ *numOverlayVisuals,* OverlayInfo ∗∗ *pOverlayVisuals,* int ∗ *numImageVisuals,* XVisualInfo ∗∗∗ *pImageVisuals*)

6.117 src/Services/MyRemoteDesktop/xwd-1.0.5/xwd.c File Reference

```
#include <stdio.h>
#include <errno.h>
#include <X11/Xos.h>
#include <stdlib.h>
#include <X11/Xlib.h>
#include <X11/Xutil.h>
#include "X11/XWDFile.h"
#include "dsimple.h"
#include "list.h"
#include "wsutils.h"
#include "multiVis.h"
```

Include dependency graph for xwd.c:

Macros

- #define FEEP_VOLUME 0
- #define lowbit(x) ((x) & (∼(x) + 1))

Typedefs

- typedef unsigned long Pixel

Functions

- int main (int, char ∗∗)
- void Window_Dump (Window, FILE ∗)
- int Image_Size (XImage ∗)
- int Get_XColors (XWindowAttributes ∗, XColor ∗∗)
- void _swapshort (register char ∗, register unsigned)
- void _swaplong (register char ∗, register unsigned)
- void usage (void)

6.117.1 Macro Definition Documentation

6.117.1.1 #define FEEP_VOLUME 0

6.117.1.2 #define lowbit(*x*) ((x) & (∼(x) + 1))

6.117.2 Typedef Documentation

6.117.2.1 typedef unsigned long **Pixel**

6.117.3 Function Documentation

6.117.3.1 void _swaplong (register char ∗ *bp,* register unsigned *n*)

6.117.3.2 void _swapshort (register char ∗ *bp,* register unsigned *n*)

6.117.3.3 int Get_XColors (XWindowAttributes ∗ *win_info,* XColor ∗∗ *colors*)

6.117.3.4 int Image_Size (XImage ∗ *image*)

6.117.3.5 int main (int *argc,* char ∗∗ *argv*)

Here is the call graph for this function:

6.117.3.6 void usage (void)

6.117.3.7 void Window_Dump (Window *window,* FILE ∗ *out*)

Here is the call graph for this function:

6.118 src/Services/MyRemoteDesktop/xwd-1.0.5/XwdLib.h File Reference

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

Functions

- int initXwdLib (int argc, char ∗∗argv)
- int closeXwdLib ()
- int getScreen (unsigned char ∗frame, unsigned int ∗frameWidth, unsigned int ∗frameHeight)

6.118.1 Function Documentation

6.118.1.1 int closeXwdLib ( )

6.118.1.2 int getScreen (unsigned char ∗ *frame,* unsigned int ∗ *frameWidth,* unsigned int ∗ *frameHeight*)

Here is the call graph for this function:

6.118.1.3 int initXwdLib (int *argc,* char ∗∗ *argv*)

Here is the call graph for this function:

6.119 src/Services/MyTube/indexer.c File Reference

```
#include "indexer.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <dirent.h>
#include <sys/stat.h>
#include <errno.h>
```

Include dependency graph for indexer.c:

Macros

- #define DEFAULT_TEST_TRANSMISSION_VIDEO_TITLE "MyTube Test Broadcast"

Functions

- char ∗ path_cat2 (const char ∗str1, const char ∗str2)
- unsigned int getAVideoForQuery (struct videoCollection ∗db, const char ∗query)
- unsigned int clearExtensionFAST (char ∗inputOutputStr)
- struct videoCollection ∗ loadVideoDatabase (char ∗directoryPath)

Variables

- unsigned int videoDefaultTestTranmission =0

6.119.1 Macro Definition Documentation

6.119.1.1 #define DEFAULT_TEST_TRANSMISSION_VIDEO_TITLE "MyTube Test Broadcast"

6.119.2 Function Documentation

6.119.2.1 unsigned int clearExtensionFAST (char ∗ *inputOutputStr*)

6.119.2.2 unsigned int getAVideoForQuery (struct videoCollection ∗ *db,* const char ∗ *query*)

6.119.2.3 struct videoCollection∗ loadVideoDatabase (char ∗ *directoryPath*)

Here is the call graph for this function:

6.119.2.4 char∗ path_cat2 (const char ∗ *str1,* const char ∗ *str2*)

6.119.3 Variable Documentation

6.119.3.1 unsigned int videoDefaultTestTranmission =0

6.120 src/Services/MyTube/indexer.h File Reference

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

Data Structures

- struct videoItem
- struct videoCollection

Macros

- #define MAX_STR 512

Functions

- char ∗ path_cat2 (const char ∗str1, const char ∗str2)
- struct videoCollection ∗ loadVideoDatabase (char ∗directoryPath)

Variables

- unsigned int videoDefaultTestTranmission

6.120.1 Macro Definition Documentation

6.120.1.1 #define MAX_STR 512

6.120.2 Function Documentation

6.120.2.1 struct **videoCollection**∗ loadVideoDatabase (char ∗ *directoryPath*)

Here is the call graph for this function:

6.120.2.2 char∗ path_cat2 (const char ∗ *str1*, const char ∗ *str2*)

6.120.3 Variable Documentation

6.120.3.1 unsigned int videoDefaultTestTranmission

6.121 src/Services/MyTube/thumbnailer.c File Reference

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

Include dependency graph for thumbnailer.c:

Functions

- char ∗ generateThumbnailOfVideo (int live, const char ∗videoDirectory, const char ∗videofile, const char ∗thumbDirectory)

6.121.1 Function Documentation

6.121.1.1 char∗ generateThumbnailOfVideo (int *live,* const char ∗ *videoDirectory,* const char ∗ *videofile,* const char ∗ *thumbDirectory*)

Here is the call graph for this function:

6.122 src/Services/MyTube/thumbnailer.h File Reference

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

Macros

- #define GENERATE_NEW_THUMBNAILS_LIVE 0

Functions

- char ∗ generateThumbnailOfVideo (int live, const char ∗videoDirectory, const char ∗videofile, const char ∗thumbDirectory)

6.122.1 Macro Definition Documentation

6.122.1.1 #define GENERATE_NEW_THUMBNAILS_LIVE 0

6.122.2 Function Documentation

6.122.2.1 char∗ generateThumbnailOfVideo (int *live,* const char ∗ *videoDirectory,* const char ∗ *videofile,* const char ∗ *thumbDirectory*)

Here is the call graph for this function:

6.123 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
- 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}

6.123.1 Macro Definition Documentation

6.123.1.1 #define ADMIN_BINDING_PORT 8082

6.123.1.2 #define DEFAULT_BINDING_PORT 8080

6.123.1.3 #define ENABLE_ADMIN_PAGE 0

6.123.1.4 #define ENABLE_CHAT_BOX 0

6.123.1.5 #define ENABLE_PASSWORD_PROTECTION 0

6.123.1.6 #define MAX_BINDING_PORT 65534

6.123.1.7 #define MAX_COMMAND_SIZE 2048

6.123.2 Function Documentation

6.123.2.1 void close_dynamic_content ( )

Here is the call graph for this function:

6.123.2.2 char EraseFile (char ∗ *filename*)

6.123.2.3 void execute (char ∗ *command,* char ∗ *param*)

bin/bash -c "

Here is the call graph for this function:

6.123.2.4 char FileExistsTest (char ∗ *filename*)

6.123.2.5 int getBackCommandLine (char ∗ *command,* char ∗ *what2GetBack,* unsigned int *what2GetBackMaxSize*)

6.123.2.6 int init_dynamic_content ( )

Here is the call graph for this function:

6.123.2.7 void joystickExecute (float *x,* float *y*)

Here is the call graph for this function:

6.123.2.8 int main (int *argc,* char ∗ *argv[ ]*)

Here is the call graph for this function:

6.123.2.9 void∗ prepare_base_image (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.123.2.10 void∗ prepare_form_content_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.123.2.11 void∗ prepare_index_content_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

6.123.2.12 void∗ prepare_stats_content_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.123.2.13 void∗ prepare_top_image (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.123.2.14 void replaceChar (char ∗ *input,* char *findChar,* char *replaceWith*)

6.123.2.15 void∗ store_new_configuration_callback (struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.123.2.16 unsigned int StringIsHTMLSafe (char ∗ *str*)

6.123.2.17 void termination_handler (int *signum*)

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

Here is the call graph for this function:

6.123.3 Variable Documentation

6.123.3.1 char admin_root[MAX_FILE_PATH] ="admin_html/"

6.123.3.2 struct AmmServer_Instance∗ admin_server =0

6.123.3.3 struct AmmServer_RH_Context base_image ={0}

6.123.3.4 struct AmmServer_RH_Context chatbox ={0}

6.123.3.5 struct AmmServer_Instance∗ default_server =0

Dynamic content code ..! START!

6.123.3.6 struct AmmServer_RH_Context form ={0}

6.123.3.7 struct AmmServer_RequestOverride_Context GET_override ={{0}}

6.123.3.8 struct AmmServer_RH_Context indexPage ={0}

6.123.3.9 char∗ page =0

6.123.3.10 unsigned int pageLength =0

6.123.3.11 struct AmmServer_RH_Context random_chars ={0}

6.123.3.12 struct AmmServer_RH_Context settings ={0}

6.123.3.13 struct AmmServer_RH_Context stats ={0}

6.123.3.14 char templates_root[MAX_FILE_PATH] ="public_html/templates/"

6.123.3.15 struct AmmServer_RH_Context top_image ={0}

6.123.3.16 char webserver_root[MAX_FILE_PATH] ="public_html/"

6.124 src/Services/SQLiteServer/sqlite.c File Reference

```
#include <sqlite3.h>
#include "sqlite.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

Include dependency graph for sqlite.c:

Functions

- int printCars (void ∗rqstV, int argc, char ∗∗argv, char ∗∗azColName)
- int SQL_init (struct SQLiteSession ∗sqlserver, const char ∗dbFilename)
- int SQL_close (struct SQLiteSession ∗sqlserver)
- int SQL_getVersion (struct SQLiteSession ∗sqlserver)
- int SQL_populate (struct SQLiteSession ∗sqlserver)
- int serveCarsPageWithSQL (struct SQLiteSession ∗sqlserver, struct AmmServer_DynamicRequest ∗rqst)

6.124.1 Function Documentation

6.124.1.1 int printCars (void ∗ *rqstV,* int *argc,* char ∗∗ *argv,* char ∗∗ *azColName*)

6.124.1.2 int serveCarsPageWithSQL (struct SQLiteSession ∗ *sqlserver,* struct AmmServer_DynamicRequest ∗ *rqst*)

Here is the call graph for this function:

6.124.1.3 int SQL_close (struct SQLiteSession ∗ *sqlserver*)

6.124.1.4 int SQL_getVersion (struct SQLiteSession ∗ *sqlserver*)

6.124.1.5 int SQL_init (struct SQLiteSession ∗ *sqlserver,* const char ∗ *dbFilename*)

Here is the call graph for this function:

6.124.1.6 int SQL_populate (struct SQLiteSession ∗ *sqlserver*)

6.125 src/Services/SQLiteServer/sqlite.h File Reference

```
#include "../../AmmServerlib/AmmServerlib.h"
```

Include dependency graph for sqlite.h: This graph shows which files directly or indirectly include this file:

Data Structures

- struct SQLiteSession

Functions

- int SQL_init (struct SQLiteSession ∗sqlserver, const char ∗dbFilename)
- int SQL_close (struct SQLiteSession ∗sqlserver)
- int SQL_getVersion (struct SQLiteSession ∗sqlserver)

- int SQL_populate (struct SQLiteSession ∗sqlserver)
- int serveCarsPageWithSQL (struct SQLiteSession ∗sqlserver, struct AmmServer_DynamicRequest ∗rqst)

6.125.1 Function Documentation

6.125.1.1 int serveCarsPageWithSQL (struct **SQLiteSession** ∗ *sqlserver,* struct **AmmServer_DynamicRequest** ∗ *rqst*)

Here is the call graph for this function:

6.125.1.2 int SQL_close (struct **SQLiteSession** ∗ *sqlserver*)

6.125.1.3 int SQL_getVersion (struct **SQLiteSession** ∗ *sqlserver*)

6.125.1.4 int SQL_init (struct **SQLiteSession** ∗ *sqlserver,* const char ∗ *dbFilename*)

Here is the call graph for this function:

6.125.1.5 int SQL_populate (struct **SQLiteSession** ∗ *sqlserver*)

6.126 src/StringRecognizer/fastStringParser.c File Reference

```
#include "fastStringParser.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <time.h>
```

Include dependency graph for fastStringParser.c:

Macros

- #define MAXIMUM_FILENAME_WITH_EXTENSION 1024
- #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 ∗resStringResultIndex, char ∗Sequence, unsigned int seqLength)
- unsigned int fastStringParser_countStringsForNextChar (struct fastStringParser ∗fsp, unsigned int ∗resStringResultIndex, char ∗Sequence, unsigned int seqLength)
- void addLevelSpaces (FILE ∗fp, unsigned int level)
- int printIfAllPossibleStrings (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 (struct fastStringParser ∗fsp)

  Destroy fast string parser.

Variables

- struct fastStringParser ∗ fspHTTPHeader = 0
- char acceptedChars [] ="ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789-_"

6.126.1 Macro Definition Documentation

6.126.1.1 #define ACTIVATED_LEVELS 3

6.126.1.2 #define MAXIMUM_FILENAME_WITH_EXTENSION 1024

6.126.1.3 #define MAXIMUM_LEVELS 123

6.126.1.4 #define MAXIMUM_LINE_LENGTH 1024

6.126.2 Function Documentation

6.126.2.1 void addLevelSpaces (FILE ∗ *fp*, unsigned int *level*)

6.126.2.2 void convertTo_ENUM_ID (char ∗ *sPtr*) [inline]

6.126.2.3 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=Success,0=Failure	

Here is the call graph for this function:

6.126.2.4 int fastStringParser_addString (struct fastStringParser ∗ *fsp*, char ∗ *str*)

Here is the call graph for this function:

6.126.2.5 int fastStringParser_close (struct fastStringParser ∗ *fsp*)

Destroy fast string parser.

Parameters

Structure	that holds the parser

Return values

1=Success,0=Failure	

6.126.2.6 unsigned int fastStringParser_countStringsForNextChar (struct **fastStringParser** ∗ *fsp,* unsigned int ∗ *resStringResultIndex,* char ∗ *Sequence,* unsigned int *seqLength*)

Here is the call graph for this function:

6.126.2.7 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:

6.126.2.8 int fastStringParser_hasStringsWithNConsecutiveChars (struct **fastStringParser** ∗ *fsp,* unsigned int ∗ *resStringResultIndex,* char ∗ *Sequence,* unsigned int *seqLength*)

6.126.2.9 struct **fastStringParser**∗ fastStringParser_initialize (unsigned int *totalStrings*)

6.126.2.10 int printAllEnumeratorItems (FILE ∗ *fp,* struct **fastStringParser** ∗ *fsp,* char ∗ *functionName*)

6.126.2.11 int printIfAllPossibleStrings (FILE ∗ *fp,* struct **fastStringParser** ∗ *fsp,* char ∗ *Sequence,* unsigned int *seqLength*)

Here is the call graph for this function:

6.126.2.12 int recursiveTraverser (FILE ∗ *fp,* struct **fastStringParser** ∗ *fsp,* char ∗ *functionName,* char ∗ *cArray,* unsigned int *level*)

Here is the call graph for this function:

6.126.3 Variable Documentation

6.126.3.1 char acceptedChars[ ] ="ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789-_"

6.126.3.2 struct **fastStringParser**∗ fspHTTPHeader = 0

6.127 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.
- int fastStringParser_close (struct fastStringParser ∗fsp)

  Destroy fast string parser.

6.127.1 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.

Bug In case the declarations have shared prefixes and the shortest prefix is stated first they will also get recognized first so be careful

Author

Ammar Qammaz (AmmarkoV)

6.127.2 Function Documentation

6.127.2.1 int export_C_Scanner (struct **fastStringParser** ∗ *fsp,* char ∗ *filename*)

Export a C Scanner source code.

Parameters

Structure	to hold all the intermediate state
Name	of the current function

Return values

1=Success,0=Failure	

Here is the call graph for this function:

6.127.2.2 int fastStringParser_close (struct **fastStringParser** ∗ *fsp*)

Destroy fast string parser.

Parameters

Structure	that holds the parser

Return values

1=Success,0=Failure	

6.127.2.3 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:

6.128 src/UnitTests/testHashMap.c File Reference

```
#include <stdio.h>
#include "../AmmServerlib/hashmap/hashmap.h"
```

Include dependency graph for testHashMap.c:

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 ∗/

Functions

- int doHashMapTest ()
- int doInjectTest ()
- int main (int argc, char ∗argv[ ])

6.128.1 Macro Definition Documentation

6.128.1.1 #define BLACK "\033[30m" /∗ Black ∗/

6.128.1.2 #define BLUE "\033[34m" /∗ Blue ∗/

6.128.1.3 #define CYAN "\033[36m" /∗ Cyan ∗/

6.128.1.4 #define GREEN "\033[32m" /∗ Green ∗/

6.128.1.5 #define MAGENTA "\033[35m" /* Magenta */

6.128.1.6 #define NORMAL "\033[0m"

6.128.1.7 #define RED "\033[31m" /* Red */

6.128.1.8 #define WHITE "\033[37m" /* White */

6.128.1.9 #define YELLOW "\033[33m" /* Yellow */

6.128.2 Function Documentation

6.128.2.1 int doHashMapTest ( )

Here is the call graph for this function:

6.128.2.2 int doInjectTest ( )

6.128.2.3 int main (int *argc,* char * *argv[ ]*)

Here is the call graph for this function:

6.129 src/UserAccounts/userAccounts.h File Reference

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

Data Structures

- struct UserAccountAuthenticationToken
- struct UserAccountDatabase

Typedefs

- typedef unsigned int UserAccount_PasswordEncoding
- typedef unsigned int UserAccount_UserID

Enumerations

- enum UserAccountPasswordEncodingEnum { ENCODING_PLAINTEXT =0, ENCODING_SHA1, ENCODING_AVAILIABLE_TYPES }

Functions

- struct UserAccountDatabase * uadb_initializeUserAccountDatabase (char *filename)
- int uadb_closeUserAccountDatabase (struct UserAccountDatabase **uadb)
- int uadb_authenticateUser (struct UserAccountDatabase *uadb, struct UserAccountAuthenticationToken *outputToken, UserAccount_UserID userID)
- int uadb_loginUser (struct UserAccountDatabase *uadb, struct UserAccountAuthenticationToken *outputToken, char *username, char *password, UserAccount_PasswordEncoding encoding, char *ip, char *browserFingerprint)

6.129.1 Typedef Documentation

6.129.1.1 typedef unsigned int UserAccount_PasswordEncoding

6.129.1.2 typedef unsigned int UserAccount_UserID

6.129.2 Enumeration Type Documentation

6.129.2.1 enum UserAccountPasswordEncodingEnum

Enumerator

ENCODING_PLAINTEXT

ENCODING_SHA1

ENCODING_AVAILIABLE_TYPES

6.129.3 Function Documentation

6.129.3.1 int uadb_authenticateUser (struct UserAccountDatabase ∗ *uadb,* struct UserAccountAuthenticationToken ∗ *outputToken,* UserAccount_UserID *userID*)

6.129.3.2 int uadb_closeUserAccountDatabase (struct UserAccountDatabase ∗∗ *uadb*)

6.129.3.3 struct UserAccountDatabase∗ uadb_initializeUserAccountDatabase (char ∗ *filename*)

6.129.3.4 int uadb_loginUser (struct UserAccountDatabase ∗ *uadb,* struct UserAccountAuthenticationToken ∗ *outputToken,* char ∗ *username,* char ∗ *password,* UserAccount_PasswordEncoding *encoding,* char ∗ *ip,* char ∗ *browserFingerprint*)

Index

~InputParser
    InputParser, 32
_FILES
    AmmServerlib.h, 121
    AmmServerlib/main.c, 64
_GET
    AmmServerlib.h, 121
    AmmServerlib/main.c, 64
_POST
    AmmServerlib.h, 122
    AmmServerlib/main.c, 64
_ipc_ver
    InputParser_C.c, 189
_list_item, 15
    curr, 15
    item, 15
    next, 15
    ptr, 15
_swaplong
    Services/MyRemoteDesktop/xwd-1.0.5/main.c, 98
    xwd.c, 298
_swapshort
    Services/MyRemoteDesktop/xwd-1.0.5/main.c, 98
    xwd.c, 298

AMMINF_ACTIVE_CLIENTS
    AmmServerlib.h, 120
AMMINF_ACTIVE_THREADS
    AmmServerlib.h, 120
AMMSET_PASSWORD_PROTECTION
    AmmServerlib.h, 120
AMMSET_PASSWORD_STR
    AmmServerlib.h, 121
AMMSET_RANDOMIZE_ETAG_BEGINNING
    AmmServerlib.h, 120
AMMSET_TEST
    AmmServerlib.h, 121
AMMSET_TESTSTR
    AmmServerlib.h, 121
AMMSET_USERNAME_STR
    AmmServerlib.h, 121
APPLICATIONFILES_CPL
    applicationFiles.h, 215
APPLICATIONFILES_DLL
    applicationFiles.h, 215
APPLICATIONFILES_EMPTY
    applicationFiles.h, 215
APPLICATIONFILES_END_OF_ITEMS
    applicationFiles.h, 215
APPLICATIONFILES_EXE
    applicationFiles.h, 215
APPLICATIONFILES_PDF
    applicationFiles.h, 215
APPLICATIONFILES_SCR
    applicationFiles.h, 215
APPLICATIONFILES_SWF
    applicationFiles.h, 215
ARCHIVEFILES_7Z
    archiveFiles.h, 218
ARCHIVEFILES_AR
    archiveFiles.h, 218
ARCHIVEFILES_BZ2
    archiveFiles.h, 218
ARCHIVEFILES_CBZ
    archiveFiles.h, 218
ARCHIVEFILES_CPIO
    archiveFiles.h, 218
ARCHIVEFILES_EMPTY
    archiveFiles.h, 218
ARCHIVEFILES_END_OF_ITEMS
    archiveFiles.h, 218
ARCHIVEFILES_GZ
    archiveFiles.h, 218
ARCHIVEFILES_ISO
    archiveFiles.h, 218
ARCHIVEFILES_JAR
    archiveFiles.h, 218
ARCHIVEFILES_LZMA
    archiveFiles.h, 218
ARCHIVEFILES_TAR
    archiveFiles.h, 218
ARCHIVEFILES_TAR_7Z
    archiveFiles.h, 218
ARCHIVEFILES_TAR_BZ
    archiveFiles.h, 218
ARCHIVEFILES_TAR_BZ2
    archiveFiles.h, 218
ARCHIVEFILES_TAR_GZ
    archiveFiles.h, 218
ARCHIVEFILES_TAR_LZ
    archiveFiles.h, 218
ARCHIVEFILES_TAR_LZMA
    archiveFiles.h, 218
ARCHIVEFILES_TAR_XZ
    archiveFiles.h, 218
ARCHIVEFILES_TAR_Z
    archiveFiles.h, 218
ARCHIVEFILES_TGZ
    archiveFiles.h, 218

ARCHIVEFILES_XZ
  archiveFiles.h, 218
ARCHIVEFILES_ZIP
  archiveFiles.h, 218
AUDIO
  http_tools.h, 257
AUDIOFILES_AU
  audioFiles.h, 220
AUDIOFILES_EMPTY
  audioFiles.h, 220
AUDIOFILES_END_OF_ITEMS
  audioFiles.h, 220
AUDIOFILES_MID
  audioFiles.h, 220
AUDIOFILES_MP3
  audioFiles.h, 220
AUDIOFILES_OGG
  audioFiles.h, 220
AUDIOFILES_VOC
  audioFiles.h, 220
AUDIOFILES_WAV
  audioFiles.h, 220
ABS
  img_warp.c, 112
ABSDIFF
  img_warp.c, 112
ACTIVATED_LEVELS
  fastStringParser.c, 307
ADMIN_BINDING_PORT
  main.cpp, 302
  ScriptRunner/main.c, 80
  Services/AmmarServer/main.c, 83
  Services/HabChan/main.c, 92
ARGC
  dsimple.c, 283
AString.c
  astringCopyOverlappingDataContent, 135
  astringInjectDataToBuffer, 135
  astringInjectDataToMemoryHandler, 135
  astringReadFileToMemory, 136
  astringReplaceAllInstancesOfVarInMemoryFile, 136
  astringReplaceVarInMemoryFile, 136
  astringWriteFileFromMemory, 136
  BLACK, 135
  GREEN, 135
  myStupidMemcpy, 136
  NORMAL, 135
  RED, 135
  YELLOW, 135
AString.h
  astringCopyOverlappingDataContent, 137
  astringInjectDataToBuffer, 137
  astringInjectDataToMemoryHandler, 137
  astringReadFileToMemory, 137
  astringReplaceAllInstancesOfVarInMemoryFile, 137
  astringReplaceVarInMemoryFile, 137
  astringWriteFileFromMemory, 137
acceptedChars
  fastStringParser.c, 308
AccessLog
  server_configuration.c, 203
  server_configuration.h, 213
AccessLogAppend
  logs.c, 262
  logs.h, 264
AccessLogEnable
  server_configuration.c, 203
  server_configuration.h, 213
active
  board, 21
Add_MyURL
  Services/MyURL/main.c, 104
add_to_list
  list.c, 287
  list.h, 290
addBoardToSite
  board.c, 269
  board.h, 269
addLevelSpaces
  fastStringParser.c, 307
addPostToThread
  state.c, 271
  state.h, 274
addThreadToBoard
  thread.c, 275
  thread.h, 275
admin_root
  main.cpp, 304
  ScriptRunner/main.c, 81
  Services/AmmarServer/main.c, 84
  Services/GeoPosShare/main.c, 90
admin_server
  main.cpp, 304
  ScriptRunner/main.c, 81
  Services/AmmarServer/main.c, 84
  state.c, 272
  state.h, 274
allocateLinksIfNeeded
  Services/MyURL/main.c, 104
allocated_links
  Services/MyURL/main.c, 105
allowComments
  website, 52
allowPing
  website, 52
AmmServerlib.h
  AMMINF_ACTIVE_CLIENTS, 120
  AMMINF_ACTIVE_THREADS, 120
  AMMSET_PASSWORD_PROTECTION, 120
  AMMSET_PASSWORD_STR, 121
  AMMSET_RANDOMIZE_ETAG_BEGINNING, 120
  AMMSET_TEST, 121
  AMMSET_TESTSTR, 121
  AMMSET_USERNAME_STR, 121

BAD, 121
CONNECT, 121
DELETE, 121
DIFFERENT_PAGE_FOR_EACH_CLIENT, 121
GET, 121
HEAD, 121
NONE, 121
OPTIONS, 121
PATCH, 121
POST, 121
PUT, 121
SAME_PAGE_FOR_ALL_CLIENTS, 121
TRACE, 121
AmmCaptcha.h
AmmCaptcha_destroy, 56
AmmCaptcha_getCaptchaFrame, 56
AmmCaptcha_getJPEGFileFromPixels, 56
AmmCaptcha_initialize, 56
AmmCaptcha_isReplyCorrect, 56
testAmmCaptcha, 57
AmmCaptcha/AmmCaptchaTester/main.c
main, 57
AmmCaptcha/main.c
AmmCaptcha_copyCaptchaJPEGImageWithCopy, 58
AmmCaptcha_destroy, 58
AmmCaptcha_getCaptchaFrame, 58
AmmCaptcha_getJPEGFileFromPixels, 58
AmmCaptcha_initialize, 58
AmmCaptcha_isReplyCorrect, 58
AmmCaptcha_loadDictionary, 58
captchaStrings, 59
convertExternalIDToInternal, 58
fontRAW, 59
fontX, 59
fontY, 59
RenderString, 59
testAmmCaptcha, 59
AmmCaptcha_copyCaptchaJPEGImageWithCopy
AmmCaptcha/main.c, 58
AmmCaptcha_destroy
AmmCaptcha.h, 56
AmmCaptcha/main.c, 58
AmmCaptcha_getCaptchaFrame
AmmCaptcha.h, 56
AmmCaptcha/main.c, 58
AmmCaptcha_getJPEGFileFromPixels
AmmCaptcha.h, 56
AmmCaptcha/main.c, 58
AmmCaptcha_initialize
AmmCaptcha.h, 56
AmmCaptcha/main.c, 58
AmmCaptcha_isReplyCorrect
AmmCaptcha.h, 56
AmmCaptcha/main.c, 58
AmmCaptcha_loadDictionary
AmmCaptcha/main.c, 58
AmmServInfos
AmmServerlib.h, 120
AmmServSettings
AmmServerlib.h, 120
AmmServStrSettings
AmmServerlib.h, 121
AmmServer_AddRequestHandler
AmmServerlib.h, 122
AmmServerlib/main.c, 64
AmmServer_AddResourceHandler
AmmServerlib.h, 122
AmmServerlib/main.c, 64
AmmServer_AllocateMemoryHandler
AmmServerlib.h, 123
AmmServerlib/main.c, 65
AmmServer_CheckIfHeaderBinaryAreTheSame
AmmServerlib.h, 123
AmmServerlib/main.c, 65
AmmServer_ConvertBufferToMemoryHandler
AmmServerlib/main.c, 65
AmmServer_CopyMemoryHandler
AmmServerlib.h, 123
AmmServerlib/main.c, 65
AmmServer_CopyOverlappingDataContent
AmmServerlib.h, 123
AmmServerlib/main.c, 65
AmmServer_DirectoryExists
AmmServerlib.h, 123
AmmServerlib/main.c, 66
AmmServer_DoNOTCacheResource
AmmServerlib.h, 124
AmmServerlib/main.c, 66
AmmServer_DoNOTCacheResourceHandler
AmmServerlib.h, 124
AmmServerlib/main.c, 66
AmmServer_DynamicRequest, 15
clientID, 16
compressedContent, 16
compressedContentSize, 16
content, 16
contentContainsPathToFileToBeStreamed, 16
contentSize, 16
GET_request, 16
GET_request_length, 16
headerResponse, 16
MAXcompressedContentSize, 16
MAXcontentSize, 16
POST_request, 16
POST_request_length, 16
AmmServer_DynamicRequestReturnFile
AmmServerlib.h, 124
AmmServerlib/main.c, 66
AmmServer_EraseFile
AmmServerlib.h, 124
AmmServerlib/main.c, 67
AmmServer_Error
AmmServerlib.h, 125
AmmServerlib/main.c, 67
AmmServer_ExecuteCommandLine

AmmServerlib.h, 125
AmmServerlib/main.c, 67
AmmServer_ExecuteCommandLineNum
AmmServerlib.h, 125
AmmServerlib/main.c, 67
AmmServer_FILES
AmmServerlib.h, 126
AmmServerlib/main.c, 68
AmmServer_FileExists
AmmServerlib.h, 126
AmmServerlib/main.c, 68
AmmServer_FileIsVideo
AmmServerlib.h, 126
AmmServerlib/main.c, 68
AmmServer_FreeMemoryHandler
AmmServerlib.h, 126
AmmServerlib/main.c, 69
AmmServer_GETArg
AmmServerlib.h, 126
AmmServerlib/main.c, 69
AmmServer_GeneralPrint
AmmServerlib/main.c, 69
AmmServer_GetInfo
AmmServerlib.h, 127
AmmServerlib/main.c, 69
AmmServer_GetIntSettingValue
AmmServerlib.h, 127
AmmServerlib/main.c, 69
AmmServer_GetStrSettingValue
AmmServerlib.h, 127
AmmServerlib/main.c, 69
AmmServer_GlobalTerminationHandler
AmmServerlib/main.c, 71
AmmServer_InjectDataToBuffer
AmmServerlib.h, 128
AmmServerlib/main.c, 71
AmmServer_Instance, 16
cache, 17
cacheHashMap, 17
cacheVersionETag, 17
clientList, 17
clientRequestHandlerOverrideContext, 17
files_open, 17
instanceName, 18
loaded_cache_items, 18
loaded_cache_items_Kbytes, 18
pause_server, 18
prespawn_jobs_finished, 18
prespawn_jobs_started, 18
prespawn_turn_to_serve, 18
prespawned_pool, 18
server_running, 18
server_thread_id, 18
serversock, 18
settings, 18
stop_server, 18
templates_root, 18
threads_pool, 18
webserver_root, 18
AmmServer_Instance_Settings, 18
BASE64PASSWORD, 19
BINDING_PORT, 19
PASSWORD, 19
USERNAME, 19
AmmServer_MemoryHandler, 19
content, 19
contentCurrentLength, 19
contentSize, 19
AmmServer_POSTArg
AmmServerlib.h, 128
AmmServerlib/main.c, 71
AmmServer_PreCacheFile
AmmServerlib/main.c, 71
AmmServer_RH_Context, 20
callback_cooldown, 21
callback_every_x_msec, 21
dynamicRequestCallbackFunction, 21
executedNow, 21
last_callback, 21
RH_Scenario, 21
requestContext, 21
resource_name, 21
web_root_path, 21
AmmServer_ReadFileToMemory
AmmServerlib.h, 128
AmmServerlib/main.c, 71
AmmServer_ReadFileToMemoryHandler
AmmServerlib.h, 128
AmmServerlib/main.c, 72
AmmServer_RegisterTerminationSignal
AmmServerlib.h, 129
AmmServerlib/main.c, 72
AmmServer_RemoveResourceHandler
AmmServerlib.h, 129
AmmServerlib/main.c, 72
AmmServer_ReplaceAllVarsInMemoryFile
AmmServerlib.h, 129
AmmServerlib/main.c, 72
AmmServer_ReplaceAllVarsInMemoryHandler
AmmServerlib.h, 130
AmmServerlib/main.c, 74
AmmServer_ReplaceCharInString
AmmServerlib.h, 130
AmmServerlib/main.c, 74
AmmServer_ReplaceVarInMemoryFile
AmmServerlib.h, 130
AmmServerlib/main.c, 74
AmmServer_ReplaceVarInMemoryHandler
AmmServerlib.h, 130
AmmServerlib/main.c, 74
AmmServer_RequestOverride_Context, 19
request, 20
request_override_callback, 20
requestHeader, 20
AmmServer_Running
AmmServerlib.h, 130

AmmServerlib/main.c, 74
AmmServer_SaveDynamicRequest
AmmServerlib.h, 131
AmmServerlib/main.c, 75
AmmServer_SelfCheck
AmmServerlib.h, 131
AmmServerlib/main.c, 75
AmmServer_SetIntSettingValue
AmmServerlib.h, 131
AmmServerlib/main.c, 75
AmmServer_SetStrSettingValue
AmmServerlib.h, 132
AmmServerlib/main.c, 76
AmmServer_SignalCountAsBadClientBehaviour
AmmServerlib.h, 132
AmmServerlib/main.c, 76
AmmServer_Start
AmmServerlib.h, 132
AmmServerlib/main.c, 76
AmmServer_StartAdminInstance
AmmServerlib.h, 132
AmmServerlib/main.c, 76
AmmServer_StartWithArgs
AmmServerlib.h, 133
AmmServerlib/main.c, 77
AmmServer_Stop
AmmServerlib.h, 133
AmmServerlib/main.c, 77
AmmServer_StringIsHTMLSafe
AmmServerlib.h, 133
AmmServerlib/main.c, 77
AmmServer_Success
AmmServerlib.h, 134
AmmServerlib/main.c, 78
AmmServer_Version
AmmServerlib.h, 134
AmmServerlib/main.c, 78
AmmServer_Warning
AmmServerlib.h, 134
AmmServerlib/main.c, 78
AmmServer_WriteFileFromMemory
AmmServerlib.h, 134
AmmServerlib/main.c, 78
AmmServerlib.h
_FILES, 121
_GET, 121
_POST, 122
AmmServInfos, 120
AmmServSettings, 120
AmmServStrSettings, 121
AmmServer_AddRequestHandler, 122
AmmServer_AddResourceHandler, 122
AmmServer_AllocateMemoryHandler, 123
AmmServer_CheckIfHeaderBinaryAreTheSame, 123
AmmServer_CopyMemoryHandler, 123
AmmServer_CopyOverlappingDataContent, 123
AmmServer_DirectoryExists, 123
AmmServer_DoNOTCacheResource, 124
AmmServer_DoNOTCacheResourceHandler, 124
AmmServer_DynamicRequestReturnFile, 124
AmmServer_EraseFile, 124
AmmServer_Error, 125
AmmServer_ExecuteCommandLine, 125
AmmServer_ExecuteCommandLineNum, 125
AmmServer_FILES, 126
AmmServer_FileExists, 126
AmmServer_FileIsVideo, 126
AmmServer_FreeMemoryHandler, 126
AmmServer_GETArg, 126
AmmServer_GetInfo, 127
AmmServer_GetIntSettingValue, 127
AmmServer_GetStrSettingValue, 127
AmmServer_InjectDataToBuffer, 128
AmmServer_POSTArg, 128
AmmServer_ReadFileToMemory, 128
AmmServer_ReadFileToMemoryHandler, 128
AmmServer_RegisterTerminationSignal, 129
AmmServer_RemoveResourceHandler, 129
AmmServer_ReplaceAllVarsInMemoryFile, 129
AmmServer_ReplaceAllVarsInMemoryHandler, 130
AmmServer_ReplaceCharInString, 130
AmmServer_ReplaceVarInMemoryFile, 130
AmmServer_ReplaceVarInMemoryHandler, 130
AmmServer_Running, 130
AmmServer_SaveDynamicRequest, 131
AmmServer_SelfCheck, 131
AmmServer_SetIntSettingValue, 131
AmmServer_SetStrSettingValue, 132
AmmServer_SignalCountAsBadClientBehaviour, 132
AmmServer_Start, 132
AmmServer_StartAdminInstance, 132
AmmServer_StartWithArgs, 133
AmmServer_Stop, 133
AmmServer_StringIsHTMLSafe, 133
AmmServer_Success, 134
AmmServer_Version, 134
AmmServer_Warning, 134
AmmServer_WriteFileFromMemory, 134
MAX_FILE_PATH, 120
MAX_QUERY, 120
MAX_RESOURCE, 120
POPEN_BUFFER_SIZE, 120
RHScenarios, 121
TypesOfRequests, 121
AmmServerlib/InputParser/InputParser_C_Tester/main.c
BLACK, 60
BLUE, 60
CYAN, 60
GREEN, 60
IntermediateTests, 60
MAGENTA, 60
main, 60

max_ret_word, 60
NORMAL, 60
ParseString, 60
RED, 60
WHITE, 60
YELLOW, 60
AmmServerlib/main.c
_FILES, 64
_GET, 64
_POST, 64
AmmServer_AddRequestHandler, 64
AmmServer_AddResourceHandler, 64
AmmServer_AllocateMemoryHandler, 65
AmmServer_CheckIfHeaderBinaryAreTheSame, 65
AmmServer_ConvertBufferToMemoryHandler, 65
AmmServer_CopyMemoryHandler, 65
AmmServer_CopyOverlappingDataContent, 65
AmmServer_DirectoryExists, 66
AmmServer_DoNOTCacheResource, 66
AmmServer_DoNOTCacheResourceHandler, 66
AmmServer_DynamicRequestReturnFile, 66
AmmServer_EraseFile, 67
AmmServer_Error, 67
AmmServer_ExecuteCommandLine, 67
AmmServer_ExecuteCommandLineNum, 67
AmmServer_FILES, 68
AmmServer_FileExists, 68
AmmServer_FileIsVideo, 68
AmmServer_FreeMemoryHandler, 69
AmmServer_GETArg, 69
AmmServer_GeneralPrint, 69
AmmServer_GetInfo, 69
AmmServer_GetIntSettingValue, 69
AmmServer_GetStrSettingValue, 69
AmmServer_GlobalTerminationHandler, 71
AmmServer_InjectDataToBuffer, 71
AmmServer_POSTArg, 71
AmmServer_PreCacheFile, 71
AmmServer_ReadFileToMemory, 71
AmmServer_ReadFileToMemoryHandler, 72
AmmServer_RegisterTerminationSignal, 72
AmmServer_RemoveResourceHandler, 72
AmmServer_ReplaceAllVarsInMemoryFile, 72
AmmServer_ReplaceAllVarsInMemoryHandler, 74
AmmServer_ReplaceCharInString, 74
AmmServer_ReplaceVarInMemoryFile, 74
AmmServer_ReplaceVarInMemoryHandler, 74
AmmServer_Running, 74
AmmServer_SaveDynamicRequest, 75
AmmServer_SelfCheck, 75
AmmServer_SetIntSettingValue, 75
AmmServer_SetStrSettingValue, 76
AmmServer_SignalCountAsBadClientBehaviour, 76
AmmServer_Start, 76
AmmServer_StartAdminInstance, 76
AmmServer_StartWithArgs, 77
AmmServer_Stop, 77
AmmServer_StringIsHTMLSafe, 77
AmmServer_Success, 78
AmmServer_Version, 78
AmmServer_Warning, 78
AmmServer_WriteFileFromMemory, 78
TerminationCallback, 78
AnalyzeHTTPHeader
http_header_analysis.c, 179
http_header_analysis.h, 182
AnalyzeHTTPLineRequest
http_header_analysis.c, 180
AnalyzePOSTLineRequest
post_header_analysis.c, 184
post_header_analysis.h, 185
android
Services/GeoPosShare/main.c, 90
apk
Services/GeoPosShare/main.c, 90
Append2MyURLDBFile
Services/MyURL/main.c, 104
appendGPS_OSM_Format
Services/GeoPosShare/main.c, 90
appendGPSMessage
Services/GeoPosShare/main.c, 90
AppendPOSTRequestToHTTPHeader
http_header_analysis.c, 180
http_header_analysis.h, 182
appendPosts
database.c, 276
applicationFiles.h
APPLICATIONFILES_CPL, 215
APPLICATIONFILES_DLL, 215
APPLICATIONFILES_EMPTY, 215
APPLICATIONFILES_END_OF_ITEMS, 215
APPLICATIONFILES_EXE, 215
APPLICATIONFILES_PDF, 215
APPLICATIONFILES_SCR, 215
APPLICATIONFILES_SWF, 215
applicationFiles.c
scanFor_applicationFiles, 214
applicationFiles.h
scanFor_applicationFiles, 216
archiveFiles.h
ARCHIVEFILES_7Z, 218
ARCHIVEFILES_AR, 218
ARCHIVEFILES_BZ2, 218
ARCHIVEFILES_CBZ, 218
ARCHIVEFILES_CPIO, 218
ARCHIVEFILES_EMPTY, 218
ARCHIVEFILES_END_OF_ITEMS, 218
ARCHIVEFILES_GZ, 218
ARCHIVEFILES_ISO, 218
ARCHIVEFILES_JAR, 218
ARCHIVEFILES_LZMA, 218
ARCHIVEFILES_TAR, 218
ARCHIVEFILES_TAR_7Z, 218
ARCHIVEFILES_TAR_BZ, 218

ARCHIVEFILES_TAR_BZ2, 218
ARCHIVEFILES_TAR_GZ, 218
ARCHIVEFILES_TAR_LZ, 218
ARCHIVEFILES_TAR_LZMA, 218
ARCHIVEFILES_TAR_XZ, 218
ARCHIVEFILES_TAR_Z, 218
ARCHIVEFILES_TGZ, 218
ARCHIVEFILES_XZ, 218
ARCHIVEFILES_ZIP, 218

archiveFiles.c
scanFor_archiveFiles, 217

archiveFiles.h
scanFor_archiveFiles, 218

AssignStr
server_configuration.c, 201
server_configuration.h, 212

astringCopyOverlappingDataContent
AString.c, 135
AString.h, 137

astringInjectDataToBuffer
AString.c, 135
AString.h, 137

astringInjectDataToMemoryHandler
AString.c, 135
AString.h, 137

astringReadFileToMemory
AString.c, 136
AString.h, 137

astringReplaceAllInstancesOfVarInMemoryFile
AString.c, 136
AString.h, 137

astringReplaceVarInMemoryFile
AString.c, 136
AString.h, 137

astringWriteFileFromMemory
AString.c, 136
AString.h, 137

audioFiles.h
AUDIOFILES_AU, 220
AUDIOFILES_EMPTY, 220
AUDIOFILES_END_OF_ITEMS, 220
AUDIOFILES_MID, 220
AUDIOFILES_MP3, 220
AUDIOFILES_OGG, 220
AUDIOFILES_VOC, 220
AUDIOFILES_WAV, 220

audioFiles.c
scanFor_audioFiles, 219

audioFiles.h
scanFor_audioFiles, 220

author
postItem, 43

authorized
HTTPHeader, 28

BAD
AmmServerlib.h, 121

BASE64PASSWORD
AmmServer_Instance_Settings, 19

BINDING_PORT
AmmServer_Instance_Settings, 19

BLACK
AmmServerlib/InputParser/InputParser_C_-Tester/main.c, 60
AString.c, 135
logs.h, 264
testHashMap.c, 310

BLUE
AmmServerlib/InputParser/InputParser_C_-Tester/main.c, 60
logs.h, 264
testHashMap.c, 310

BLUE_SHIFT
multiVis.c, 293

BOLDBLACK
logs.h, 264

BOLDBLUE
logs.h, 264

BOLDCYAN
logs.h, 264

BOLDGREEN
logs.h, 264

BOLDMAGENTA
logs.h, 264

BOLDRED
logs.h, 264

BOLDWHITE
logs.h, 264

BOLDYELLOW
logs.h, 264

base_image
main.cpp, 304
ScriptRunner/main.c, 81

bitBltImage
imaging.c, 111
imaging.h, 112

bitBltImageRotated
imaging.c, 111

blogTitle
website, 52

board, 21
active, 21
currentThreads, 21
currentUsers, 21
hidden, 21
imageUID, 21
maxThreads, 22
name, 22
postUID, 22
threadQueue, 22
threadUID, 22
threads, 22

board.c
addBoardToSite, 269
loadBoardSettings, 269
prepareBoardIndexView, 269

board.h

addBoardToSite, 269
prepareBoardIndexView, 269
boardHashMap
state.c, 272
state.h, 274
boardIndexView
Services/HabChan/main.c, 92
boards
site, 45
border
image_region_type, 31
border_width
image_win_type, 31
boundary
HTTPHeader, 28
boundaryLength
HTTPHeader, 28
busy
PreSpawnedThread, 44

CMD_TYPE_BELL_OFF
Services/CinemaPilot/main.c, 86
CMD_TYPE_BELL_ON
Services/CinemaPilot/main.c, 86
CMD_TYPE_INTERMISSION
Services/CinemaPilot/main.c, 86
CMD_TYPE_LIGHTS_OFF
Services/CinemaPilot/main.c, 86
CMD_TYPE_LIGHTS_ON
Services/CinemaPilot/main.c, 86
CMD_TYPE_MOVIE
Services/CinemaPilot/main.c, 86
CMD_TYPE_NONE
Services/CinemaPilot/main.c, 86
CMD_TYPE_SOUND_OFF
Services/CinemaPilot/main.c, 86
CMD_TYPE_SOUND_ON
Services/CinemaPilot/main.c, 86
CMD_TYPE_TRAILER
Services/CinemaPilot/main.c, 86
CONNECT
AmmServerlib.h, 121
CACHING_ENABLED
server_configuration.c, 203
server_configuration.h, 213
CHANGE_PRIORITY
server_configuration.c, 203
server_configuration.h, 213
CHANGE_TO_UID
server_configuration.c, 203
server_configuration.h, 213
CONTAINERS_MAX
InputParser_C.h, 191
CONTENT_BUFFER
database.h, 277
COPYOPT
dsimple.c, 283
CR
http_header_analysis.c, 179
CYAN
AmmServerlib/InputParser/InputParser_C_-Tester/main.c, 60
logs.h, 264
testHashMap.c, 310
cache
AmmServer_Instance, 17
cache_AddDoNOTCacheRuleForResource
file_caching.c, 148
file_caching.h, 156
cache_AddFile
file_caching.c, 148
file_caching.h, 156
cache_AddMemoryBlock
file_caching.c, 149
file_caching.h, 156
cache_ChangeRequestIfTemplateRequested
file_caching.c, 149
file_caching.h, 157
cache_CountMemoryUsageAllocateOperation
file_caching.c, 149
file_caching.h, 157
cache_CountMemoryUsageFreeOperation
file_caching.c, 150
file_caching.h, 157
cache_CreateResource
file_caching.c, 150
cache_Destroy
file_caching.c, 150
file_caching.h, 158
cache_DestroyResource
file_caching.c, 150
cache_FindResource
file_caching.c, 150
file_caching.h, 158
cache_GetHashOfResource
file_caching.c, 151
file_caching.h, 158
cache_GetResource
file_caching.c, 151
file_caching.h, 158
cache_Initialize
file_caching.c, 151
file_caching.h, 159
cache_LoadResourceFromDisk
file_caching.c, 153
cache_RandomizeETAG
file_caching.c, 153
file_caching.h, 159
cache_RefreshResource
file_caching.c, 153
cache_RemoveContextAndResource
file_caching.c, 153
file_caching.h, 159
cache_RemoveResource
file_caching.c, 153
file_caching.h, 161
cache_ResourceExists

file_caching.c, 154
file_caching.h, 161
cache_item, 22
compressedContent, 22
compressedContentSize, 22
content, 22
contentSize, 22
contentTypeID, 22
doNOTCacheRule, 22
dynamicRequest, 23
dynamicRequestCallbackFunction, 23
modification, 23
cacheHashMap
AmmServer_Instance, 17
cacheVersionETag
AmmServer_Instance, 17
callClientRequestHandler
dynamic_requests.c, 142
dynamic_requests.h, 145
callback_cooldown
AmmServer_RH_Context, 21
callback_every_x_msec
AmmServer_RH_Context, 21
captcha_url
Services/MyURL/main.c, 105
captchaStrings
AmmCaptcha/main.c, 59
chatbox
main.cpp, 304
ScriptRunner/main.c, 81
Services/AmmarServer/main.c, 84
CheckDelimeterNumOk
InputParser_C.c, 187
CheckHTTPHeaderCategory
http_tools.c, 250
http_tools.h, 258
CheckHTTPHeaderCategoryAllCaps
http_tools.c, 250
http_tools.h, 258
CheckIPCOk
InputParser_C.c, 187
CheckIfFileIsVideo
http_tools.c, 250
http_tools.h, 258
CheckWordNumOk
InputParser_C.c, 187
InputParser_C.h, 191
checksum
guard_byte, 25
clearExtensionFAST
indexer.c, 299
clearItemCallbackFunction
hashMap, 25
client
PassToHTTPThread, 39
PreSpawnedThread, 44
client_list.c
clientList_GetClientId, 138
clientList_close, 138
clientList_initialize, 138
clientList_isClientAllowedToMakeAConnection, 139
clientList_isClientAllowedToUseResource, 139
clientList_isClientBanned, 139
clientList_signalClientStoppedUsingResource, 139
client_list.h
clientID, 140
clientList_GetClientId, 141
clientList_close, 140
clientList_initialize, 141
clientList_isClientAllowedToMakeAConnection, 141
clientList_isClientAllowedToUseResource, 141
clientList_isClientBanned, 141
clientList_signalClientStoppedUsingResource, 142
clientID
AmmServer_DynamicRequest, 16
client_list.h, 140
clientList
AmmServer_Instance, 17
clientList_GetClientId
client_list.c, 138
client_list.h, 141
clientList_close
client_list.c, 138
client_list.h, 140
clientList_initialize
client_list.c, 138
client_list.h, 141
clientList_isClientAllowedToMakeAConnection
client_list.c, 139
client_list.h, 141
clientList_isClientAllowedToUseResource
client_list.c, 139
client_list.h, 141
clientList_isClientBanned
client_list.c, 139
client_list.h, 141
clientList_signalClientStoppedUsingResource
client_list.c, 139
client_list.h, 142
clientListContext, 23
userList, 23
clientListID
HTTPTransaction, 29
clientRequestHandlerOverrideContext
AmmServer_Instance, 17
clientServer.c
ServeClient, 233
ServeClientKeepAliveLoop, 234
clientServer.h
ServeClient, 234
clientSock
HTTPTransaction, 29
clientlen
PassToHTTPThread, 39

PreSpawnedThread, 44
clientsock
PassToHTTPThread, 39
PreSpawnedThread, 44
clientwin.c
Find_Client, 281
clientwin.h
Find_Client, 281
Close_Display
dsimple.c, 284
dsimple.h, 285
close_dynamic_content
helloworld.c, 55
main.cpp, 303
ScriptRunner/main.c, 80
Services/AmmarServer/main.c, 83
Services/CinemaPilot/main.c, 87
Services/GeoPosShare/main.c, 90
Services/HabChan/main.c, 92
Services/MyBlog/main.c, 93
Services/MyLoader/main.c, 94
Services/MyRemoteDesktop/main.c, 96
Services/MyTube/main.c, 100
Services/MyURL/main.c, 104
Services/SimpleTemplate/main.c, 107
Services/SQLiteServer/main.c, 108
closeXwdLib
Services/MyRemoteDesktop/xwd-1.0.5/main.c, 98
XwdLib.h, 298
cmap
image_region_type, 31
image_win_type, 31
cmpHashTableItems
hashmap.c, 166
command
playlistItem, 41
commandContext
Services/MyRemoteDesktop/main.c, 97
commandType
Services/CinemaPilot/main.c, 86
comment
videoItem, 52
compressedContent
AmmServer_DynamicRequest, 16
cache_item, 22
compressedContentSize
AmmServer_DynamicRequest, 16
cache_item, 22
config.h
HAVE_INTTYPES_H, 282
HAVE_MEMORY_H, 282
HAVE_STDINT_H, 282
HAVE_STDLIB_H, 282
HAVE_STRING_H, 282
HAVE_STRINGS_H, 282
HAVE_SYS_STAT_H, 282
HAVE_SYS_TYPES_H, 282
HAVE_UNISTD_H, 282
PACKAGE, 282
PACKAGE_BUGREPORT, 282
PACKAGE_NAME, 282
PACKAGE_STRING, 282
PACKAGE_TARNAME, 282
PACKAGE_URL, 282
PACKAGE_VERSION, 282
STDC_HEADERS, 282
VERSION, 282
container_end
InputParserC, 34
container_start
InputParserC, 34
content
AmmServer_DynamicRequest, 16
AmmServer_MemoryHandler, 19
cache_item, 22
postItem, 43
widgetItem, 53
contentContainsPathToFileToBeStreamed
AmmServer_DynamicRequest, 16
contentCurrentLength
AmmServer_MemoryHandler, 19
contentDisposition
HTTPHeader, 28
contentDispositionLength
HTTPHeader, 28
ContentLength
HTTPHeader, 28
contentSize
AmmServer_DynamicRequest, 16
AmmServer_MemoryHandler, 19
cache_item, 22
contentType
http_tools.h, 257
HTTPHeader, 28
contentTypeEnumerator
http_tools.h, 257
contentTypeID
cache_item, 22
contentTypeLength
HTTPHeader, 28
contents
fastStringParser, 24
convertExternalIDToInternal
AmmCaptcha/main.c, 58
convertTo_ENUM_ID
fastStringParser.c, 307
convertToUpperCase
http_tools.c, 251
cookie
HTTPHeader, 28
cookieLength
HTTPHeader, 28
coolPHPWave
img_warp.c, 112
img_warp.h, 113
copyImage

imaging.c, 111
imaging.h, 112
create_url
Services/MyURL/main.c, 105
CreateCompressedVersionofCachedResource
file_compression.c, 162
file_compression.h, 164
CreateCompressedVersionofDynamicContent
file_compression.c, 162
file_compression.h, 164
CreateCompressedVersionofStaticContent
file_compression.c, 163
file_compression.h, 164
CreateCompressedVersionofStaticContentPreloading
file_compression.c, 163
file_compression.h, 165
createImage
imaging.c, 111
imaging.h, 112
CreateImagePlanesWindow
wsutils.h, 296
CreateOverlayPlanesWindow
wsutils.h, 296
creation
post, 42
thread, 47
cur_container_count
InputParserC, 34
cur_delimeter_count
InputParserC, 34
curNumberOfEntries
hashMap, 25
curr
_list_item, 15
currentDataLength
htmlContent, 27
currentItems
linkItemList, 35
menuItemList, 36
widgetItemList, 54
currentPosts
postItemList, 43
currentTags
tagItemList, 47
currentThreads
board, 21
currentUsers
board, 21

DELETE
AmmServerlib.h, 121
DIFFERENT_PAGE_FOR_EACH_CLIENT
AmmServerlib.h, 121
DEFAULT_BINDING_PORT
main.cpp, 302
DELIM_MAX_MAX
InputParser_C.h, 191
DESTRUCT_FUNC_PTR
list.h, 290
DIRECT_COLOR
multiVis.c, 293
DISPLAY_DEBUG_INFO
imaging.c, 111
DUP_WHOLE_LIST
list.h, 289
data
htmlContent, 27
database.c
appendPosts, 276
loadPostsFromSQL, 276
myblog, 276
SQL_close, 276
SQL_createInitialTables, 276
SQL_error, 276
SQL_getVersion, 276
SQL_init, 276
sqlserver, 276
database.h
CONTENT_BUFFER, 277
loadPostsFromSQL, 278
MAX_CONTENT, 277
MAX_MENU_ITEMS, 277
MAX_STR, 277
MAX_TAGS_PER_POST, 277
MAX_WIDGET_ITEMS, 278
myblog, 278
SQL_close, 278
SQL_createInitialTables, 278
SQL_init, 278
sqlserver, 278
database_root
Services/MyTube/main.c, 101
dateStr
postItem, 43
day
timestamp, 49
days
time_provider.c, 266
db
SQLiteSession, 46
db_addIDLock
Services/MyURL/main.c, 105
db_file
Services/MyURL/main.c, 105
db_fileLock
Services/MyURL/main.c, 105
debug_get_callback
Services/AmmarServer/main.c, 83
state.c, 271
default_failed
Services/MyURL/main.c, 106
default_server
main.cpp, 304
ScriptRunner/main.c, 81
Services/AmmarServer/main.c, 84
Services/CinemaPilot/main.c, 88
Services/GeoPosShare/main.c, 90

Services/MyBlog/main.c, 93
Services/MyLoader/main.c, 95
Services/MyRemoteDesktop/main.c, 97
Services/MyTube/main.c, 101
Services/SimpleTemplate/main.c, 107
Services/SQLiteServer/main.c, 109
state.c, 272
state.h, 274
DefaultDelimeterSetup
InputParser, 33
delete_from_list
list.c, 287
list.h, 290
delete_list
list.c, 287
list.h, 290
delete_list_destroying
list.c, 287
list.h, 290
delimeters
InputParserC, 34
depth
Image, 30
destroy_index_prototype
index.c, 279
index.h, 279
destroyImage
imaging.c, 111
imaging.h, 112
difference
time_snap, 48
directory_lists.c
ending, 247
GenerateDirectoryPage, 247
path_cat, 247
starting, 247
tag_after_image, 247
tag_pre_image, 247
directory_lists.h
GenerateDirectoryPage, 248
DirectoryExistsAmmServ
http_tools.c, 251
http_tools.h, 258
dislikes
videoItem, 52
doHashMapTest
testHashMap.c, 311
doInjectTest
testHashMap.c, 311
doNOTCacheRule
cache_item, 22
doc/DoxygenMainpage.h, 55
doc/helloworld.c, 55
dpy
dsimple.c, 284
dsimple.h, 286
dsimple.c
ARGC, 283
COPYOPT, 283
Close_Display, 284
dpy, 284
Fatal_Error, 284
Get_Display_Name, 284
getRootWindow, 284
NXTOPT, 283
NXTOPTP, 283
OPTION, 283
Open_Display, 284
outl, 284
program_name, 284
screen, 284
Select_Window, 284
Select_Window_Args, 284
Setup_Display_And_Screen, 284
Setup_Null_Display_And_Screen, 284
Window_With_Name, 284
dsimple.h
Close_Display, 285
dpy, 286
Fatal_Error, 285
Get_Display_Name, 285
getRootWindow, 286
INIT_NAME, 285
MAX, 285
MIN, 285
Open_Display, 286
outl, 286
program_name, 286
screen, 286
Select_Window, 286
Select_Window_Args, 286
Setup_Display_And_Screen, 286
Setup_Null_Display_And_Screen, 286
usage, 286
Window_With_Name, 286
X_USAGE, 285
dummy
UserAccountAuthenticationToken, 50
UserAccountDatabase, 51
dup_list_head
list.c, 287
list.h, 290
dynamic_requests.c
callClientRequestHandler, 142
dynamicRequest_ContentAvailiable, 144
dynamicRequest_serveContent, 144
saveDynamicRequest, 144
dynamic_requests.h
callClientRequestHandler, 145
dynamicRequest_ContentAvailiable, 146
dynamicRequest_serveContent, 146
saveDynamicRequest, 146
dynamicRequest
cache_item, 23
dynamicRequest_ContentAvailiable
dynamic_requests.c, 144

dynamic_requests.h, 146
dynamicRequest_serveContent
dynamic_requests.c, 144
dynamic_requests.h, 146
dynamicRequestCallbackFunction
AmmServer_RH_Context, 21
cache_item, 23

ENCODING_AVAILIABLE_TYPES
userAccounts.h, 312
ENCODING_PLAINTEXT
userAccounts.h, 312
ENCODING_SHA1
userAccounts.h, 312
EXECUTABLE
http_tools.h, 258
ENABLE_ADMIN_PAGE
main.cpp, 302
ScriptRunner/main.c, 80
Services/AmmarServer/main.c, 83
Services/GeoPosShare/main.c, 89
ENABLE_CHAT_BOX
main.cpp, 302
ScriptRunner/main.c, 80
Services/AmmarServer/main.c, 83
ENABLE_COMPRESSION
server_configuration.h, 208
ENABLE_POST
server_configuration.h, 209
ENABLE_STOP_PAGE
Services/AmmarServer/main.c, 83
EQUAL
list.h, 289
eTag
HTTPHeader, 28
eTagLength
HTTPHeader, 28
EmmitPossibleConfigurationWarnings
server_configuration.c, 201
server_configuration.h, 212
empty_buffer
jpgInput.c, 114
encodeToBase64
http_tools.c, 251
http_tools.h, 258
end_timer
time_provider.c, 265
time_provider.h, 267
ending
directory_lists.c, 247
entries
hashMap, 25
entryAllocationStep
hashMap, 25
EraseFile
main.cpp, 303
err_msg
SQLiteSession, 46
error
logs.c, 262
logs.h, 264
error_url
Services/MyURL/main.c, 106
ErrorLog
server_configuration.c, 203
server_configuration.h, 213
ErrorLogAppend
logs.c, 263
logs.h, 265
ErrorLogEnable
server_configuration.c, 203
server_configuration.h, 213
execute
main.cpp, 303
ScriptRunner/main.c, 80
executePlaylist
Services/CinemaPilot/main.c, 87
executePlaylistCurrentItem
Services/CinemaPilot/main.c, 87
executeScript
Services/AmmarServer/main.c, 84
executeScriptFunction
Services/AmmarServer/main.c, 83
executeScriptRC
Services/AmmarServer/main.c, 84
executedNow
AmmServer_RH_Context, 21
export_C_Scanner
fastStringParser.c, 307
fastStringParser.h, 309
extents
my_XRegion, 36

FILETYPE_AUDIO
state.h, 273
FILETYPE_FORBIDDEN
state.h, 273
FILETYPE_IMAGE
state.h, 273
FILETYPE_VIDEO_FILE
state.h, 273
FILETYPE_VIDEO_YOUTUBE
state.h, 273
FIRSTLINES_CONNECT
firstLines.h, 222
FIRSTLINES_DELETE
firstLines.h, 222
FIRSTLINES_EMPTY
firstLines.h, 221
FIRSTLINES_END_OF_ITEMS
firstLines.h, 222
FIRSTLINES_GET
firstLines.h, 221
FIRSTLINES_HEAD
firstLines.h, 221
FIRSTLINES_OPTIONS
firstLines.h, 222
FIRSTLINES_PATCH

firstLines.h, 222
FIRSTLINES_POST
firstLines.h, 221
FIRSTLINES_PUT
firstLines.h, 221
FIRSTLINES_TRACE
firstLines.h, 222
FOLDER
http_tools.h, 258
FEEP_VOLUME
Services/MyRemoteDesktop/xwd-1.0.5/main.c, 98
xwd.c, 298
FILETYPES_ENUM
state.h, 273
FLEXIBLE
wsutils.h, 296
facebookURL
socialLinks, 45
fastJPGHeaderCheck
jpgInput.c, 114
fastSTringParser_createRulesFromFile
fastStringParser.c, 308
fastStringParser.h, 310
fastStringParser, 23
contents, 24
functionName, 24
longestStringLength, 24
MAXstringsLoaded, 24
shortestStringLength, 24
stringsLoaded, 24
fastStringParser.c
ACTIVATED_LEVELS, 307
acceptedChars, 308
addLevelSpaces, 307
convertTo_ENUM_ID, 307
export_C_Scanner, 307
fastSTringParser_createRulesFromFile, 308
fastStringParser_addString, 307
fastStringParser_close, 307
fastStringParser_countStringsForNextChar, 308
fastStringParser_hasStringsWithNConsecutiveChars, 308
fastStringParser_initialize, 308
fspHTTPHeader, 308
MAXIMUM_LEVELS, 307
printAllEnumeratorItems, 308
printIfAllPossibleStrings, 308
recursiveTraverser, 308
fastStringParser.h
export_C_Scanner, 309
fastSTringParser_createRulesFromFile, 310
fastStringParser_close, 309
fastStringParser_addString
fastStringParser.c, 307
fastStringParser_close
fastStringParser.c, 307
fastStringParser.h, 309
fastStringParser_countStringsForNextChar
fastStringParser.c, 308
fastStringParser_hasStringsWithNConsecutiveChars
fastStringParser.c, 308
fastStringParser_initialize
fastStringParser.c, 308
Fatal_Error
dsimple.c, 284
dsimple.h, 285
favicon
Services/MyTube/main.c, 101
faviconContext
Services/MyTube/main.c, 101
file_caching.c
cache_AddDoNOTCacheRuleForResource, 148
cache_AddFile, 148
cache_AddMemoryBlock, 149
cache_ChangeRequestIfTemplateRequested, 149
cache_CountMemoryUsageAllocateOperation, 149
cache_CountMemoryUsageFreeOperation, 150
cache_CreateResource, 150
cache_Destroy, 150
cache_DestroyResource, 150
cache_FindResource, 150
cache_GetHashOfResource, 151
cache_GetResource, 151
cache_Initialize, 151
cache_LoadResourceFromDisk, 153
cache_RandomizeETAG, 153
cache_RefreshResource, 153
cache_RemoveContextAndResource, 153
cache_RemoveResource, 153
cache_ResourceExists, 154
freeMallocIfNeeded, 154
file_caching.h
cache_AddDoNOTCacheRuleForResource, 156
cache_AddFile, 156
cache_AddMemoryBlock, 156
cache_ChangeRequestIfTemplateRequested, 157
cache_CountMemoryUsageAllocateOperation, 157
cache_CountMemoryUsageFreeOperation, 157
cache_Destroy, 158
cache_FindResource, 158
cache_GetHashOfResource, 158
cache_GetResource, 158
cache_Initialize, 159
cache_RandomizeETAG, 159
cache_RemoveContextAndResource, 159
cache_RemoveResource, 161
cache_ResourceExists, 161
freeMallocIfNeeded, 161
file_compression.c
CreateCompressedVersionofCachedResource, 162
CreateCompressedVersionofDynamicContent, 162
CreateCompressedVersionofStaticContent, 163

CreateCompressedVersionofStaticContentPreloading
163
file_compression.h
CreateCompressedVersionofCachedResource, 164
CreateCompressedVersionofDynamicContent, 164
CreateCompressedVersionofStaticContent, 164
CreateCompressedVersionofStaticContentPreloading, 165
file_server.c
files_open, 195
SendErrorFile, 194
SendFile, 194
SendMemoryBlockAsFile, 195
SendPart, 195
TransmitFileToSocket, 195
TransmitFileToSocketInternal, 195
file_server.h
SendErrorFile, 196
SendFile, 196
SendMemoryBlockAsFile, 197
fileCachedName
post, 42
fileDimensionHeight
post, 42
fileDimensionWidth
post, 42
FileExistsAmmServ
http_tools.c, 251
http_tools.h, 258
FileExistsTest
main.cpp, 303
fileOriginalName
post, 42
fileType
post, 42
filename
videoItem, 52
FilenameStripperOk
http_tools.c, 251
http_tools.h, 259
files_open
AmmServer_Instance, 17
file_server.c, 195
Find_Client
clientwin.c, 281
clientwin.h, 281
Find_longURL
Services/MyURL/main.c, 104
Find_longURLSerial
Services/MyURL/main.c, 104
FindAProperThreadID
freshThreads.c, 235
FindImagePlanesVisual
wsutils.h, 297
FindIndexFile
http_tools.c, 252
http_tools.h, 259
findOutClientIDOfPeer
http_tools.c, 252
http_tools.h, 259
FindOverlayPlanesVisual
wsutils.h, 297
firstLines.h
FIRSTLINES_CONNECT, 222
FIRSTLINES_DELETE, 222
FIRSTLINES_EMPTY, 221
FIRSTLINES_END_OF_ITEMS, 222
FIRSTLINES_GET, 221
FIRSTLINES_HEAD, 221
FIRSTLINES_OPTIONS, 222
FIRSTLINES_PATCH, 222
FIRSTLINES_POST, 221
FIRSTLINES_PUT, 221
FIRSTLINES_TRACE, 222
first_in_list
list.c, 287
list.h, 291
firstLines.c
scanFor_firstLines, 221
firstLines.h
scanFor_firstLines, 222
fontRAW
AmmCaptcha/main.c, 59
fontX
AmmCaptcha/main.c, 59
fontY
AmmCaptcha/main.c, 59
form
main.cpp, 304
ScriptRunner/main.c, 81
Services/AmmarServer/main.c, 84
frame_only
Services/MyRemoteDesktop/xwd-1.0.5/main.c, 99
FreeHTTPHeader
http_header_analysis.c, 180
http_header_analysis.h, 183
freeMallocIfNeeded
file_caching.c, 154
file_caching.h, 161
freeString
http_tools.c, 252
http_tools.h, 259
FreeXVisualInfo
multiVis.c, 294
wsutils.h, 297
fresh
Services/AmmarServer/main.c, 84
freshThreads.c
FindAProperThreadID, 235
SpawnThreadToServeNewClient, 235
freshThreads.h
SpawnThreadToServeNewClient, 236
fspHTTPHeader
fastStringParser.c, 308
fspString, 24

str, 24
strIDFriendly, 24
strLength, 24
fullScreenViewerPath
Services/CinemaPilot/main.c, 88
functionName
fastStringParser, 24

GET
AmmServerlib.h, 121
GET_override
main.cpp, 304
ScriptRunner/main.c, 81
Services/AmmarServer/main.c, 84
Services/CinemaPilot/main.c, 88
Services/GeoPosShare/main.c, 90
Services/MyBlog/main.c, 93
Services/MyLoader/main.c, 95
Services/MyRemoteDesktop/main.c, 97
Services/MyTube/main.c, 101
Services/SimpleTemplate/main.c, 107
Services/SQLiteServer/main.c, 109
state.c, 272
state.h, 274
GET_request
AmmServer_DynamicRequest, 16
GET_request_length
AmmServer_DynamicRequest, 16
GETquery
HTTPHeader, 28
GRAY_SCALE
multiVis.c, 293
GREATER
list.h, 289
GREEN
AmmServerlib/InputParser/InputParser_C_-Tester/main.c, 60
AString.c, 135
logs.h, 264
testHashMap.c, 310
GREEN_SHIFT
multiVis.c, 293
GenerateDirectoryPage
directory_lists.c, 247
directory_lists.h, 248
generateThumbnailOfVideo
thumbnailer.c, 301
thumbnailer.h, 301
Get_Display_Name
dsimple.c, 284
dsimple.h, 285
Get_XColors
Services/MyRemoteDesktop/xwd-1.0.5/main.c, 98
xwd.c, 298
Get_longURL
Services/MyURL/main.c, 104
getAVideoForQuery
indexer.c, 299
getBackCommandLine
main.cpp, 303
ScriptRunner/main.c, 80
GetContentType
http_tools.c, 252
http_tools.h, 259
GetContentTypeForExtension
http_tools.c, 252
GetDateString
time_provider.c, 266
time_provider.h, 267
GetDelimeter
InputParser, 33
GetExtensionImage
http_tools.c, 252
http_tools.h, 260
GetExtentionType
http_tools.c, 254
http_tools.h, 260
getFooterLinksHTML
index.c, 279
GetIntFromHTTPHeaderFieldPayload
http_tools.c, 254
http_tools.h, 260
getLeftBlogRollHTML
index.c, 279
GetLowercaseWord
InputParser, 33
getMenuListHTML
index.c, 279
GetMultiVisualRegions
multiVis.c, 294
multiVis.h, 295
GetNewStringFromHTTPHeaderFieldPayload
http_tools.c, 254
http_tools.h, 260
getPostListHTML
index.c, 279
getRightBlogRollHTML
index.c, 279
getRootWindow
dsimple.c, 284
dsimple.h, 286
getScreen
Services/MyRemoteDesktop/xwd-1.0.5/main.c, 98
XwdLib.h, 298
GetTickCountAmmServ
time_provider.c, 266
time_provider.h, 268
GetUpcaseWord
InputParser, 33
getWidgetListHTML
index.c, 279
GetWord
InputParser, 33
GetWordChar
InputParser, 33
GetWordInt
InputParser, 33

GetWordLength
InputParser, 33
GetXVisualInfo
multiVis.c, 294
wsutils.h, 297
getdbg
Services/AmmarServer/main.c, 84
goto_url
Services/MyURL/main.c, 106
gps
Services/AmmarServer/main.c, 84
Services/GeoPosShare/main.c, 91
guard_byte, 25
checksum, 25
guardbyte1
InputParserC, 34
guardbyte2
InputParserC, 34
guardbyte3
InputParserC, 34
guardbyte4
InputParserC, 34

HEAD
AmmServerlib.h, 121
HTTPHEADER_ACCEPT_ENCODING
httpHeader.h, 223
HTTPHEADER_AUTHORIZATION
httpHeader.h, 223
HTTPHEADER_CONNECTION
httpHeader.h, 223
HTTPHEADER_COOKIE
httpHeader.h, 223
HTTPHEADER_EMPTY
httpHeader.h, 223
HTTPHEADER_END_OF_ITEMS
httpHeader.h, 224
HTTPHEADER_HOST
httpHeader.h, 223
HTTPHEADER_IF_MODIFIED_SINCE
httpHeader.h, 223
HTTPHEADER_IF_NONE_MATCH
httpHeader.h, 223
HTTPHEADER_RANGE
httpHeader.h, 223
HTTPHEADER_REFERER
httpHeader.h, 224
HTTPHEADER_REFERRER
httpHeader.h, 223
HTTPHEADER_USER_AGENT
httpHeader.h, 224
HAVE_INTTYPES_H
config.h, 282
HAVE_MEMORY_H
config.h, 282
HAVE_STDINT_H
config.h, 282
HAVE_STDLIB_H
config.h, 282
HAVE_STRING_H
config.h, 282
HAVE_STRINGS_H
config.h, 282
HAVE_SYS_STAT_H
config.h, 282
HAVE_SYS_TYPES_H
config.h, 282
HAVE_UNISTD_H
config.h, 282
HTTPHeader, 27
authorized, 28
boundary, 28
boundaryLength, 28
contentDisposition, 28
contentDispositionLength, 28
ContentLength, 28
contentType, 28
contentTypeLength, 28
cookie, 28
cookieLength, 28
eTag, 28
eTagLength, 28
GETquery, 28
headerRAW, 28
headerRAWSize, 28
host, 28
hostLength, 28
keepalive, 28
POSTrequest, 28
POSTrequestSize, 28
range_end, 28
range_start, 28
referer, 28
refererLength, 28
requestType, 28
resource, 29
supports_compression, 29
userAgent, 29
userAgentLength, 29
verified_local_resource, 29
HTTPHeaderComplete
http_header_analysis.c, 180
http_header_analysis.h, 183
HTTPHeaderIsPOST
http_header_analysis.c, 180
http_header_analysis.h, 183
HTTPServerIsRunning
threadedServer.c, 241
threadedServer.h, 244
HTTPTransaction, 29
clientListID, 29
clientSock, 29
incomingHeader, 29
instance, 29
outgoingBody, 29
outgoingBodySize, 29
prespawnedThreadFlag, 29

resourceCacheID, 30
threadID, 30
hasFile
post, 42
hashFunction
hashmap.c, 166
hashmap.h, 173
hashID
videoItem, 52
hashMap, 25
clearItemCallbackFunction, 25
curNumberOfEntries, 25
entries, 25
entryAllocationStep, 25
hm_addLock, 25
hm_fileLock, 25
maxNumberOfEntries, 25
hashMap_Add
hashmap.c, 166
hashmap.h, 173
hashMap_AddULong
hashmap.c, 166
hashmap.h, 173
hashMap_Clear
hashmap.c, 167
hashmap.h, 173
hashMap_ContainsKey
hashmap.c, 167
hashmap.h, 173
hashMap_ContainsValue
hashmap.c, 167
hashmap.h, 175
hashMap_Create
hashmap.c, 167
hashmap.h, 175
hashMap_Destroy
hashmap.c, 168
hashmap.h, 175
hashMap_FindIndex
hashmap.c, 168
hashmap.h, 175
hashMap_GetCurrentNumberOfEntries
hashmap.c, 168
hashmap.h, 176
hashMap_GetHashAtIndex
hashmap.c, 168
hashmap.h, 176
hashMap_GetKeyAtIndex
hashmap.c, 169
hashmap.h, 176
hashMap_GetMaxNumberOfEntries
hashmap.c, 169
hashmap.h, 176
hashMap_GetPayload
hashmap.c, 169
hashmap.h, 177
hashMap_GetULongPayload
hashmap.c, 169
hashmap.h, 177
hashMap_Grow
hashmap.c, 170
hashMap_IsOK
hashmap.c, 170
hashMap_IsSorted
hashmap.c, 170
hashMap_LoadToFile
hashmap.c, 170
hashmap.h, 177
hashMap_SaveToFile
hashmap.c, 170
hashmap.h, 177
hashMap_Sort
hashmap.c, 170
hashmap.h, 178
hashMapEntry, 26
hits, 26
key, 26
keyHash, 26
keyLength, 26
payload, 26
payloadLength, 26
hashURL
Services/MyURL/main.c, 104
hashmap.c
cmpHashTableItems, 166
hashFunction, 166
hashMap_Add, 166
hashMap_AddULong, 166
hashMap_Clear, 167
hashMap_ContainsKey, 167
hashMap_ContainsValue, 167
hashMap_Create, 167
hashMap_Destroy, 168
hashMap_FindIndex, 168
hashMap_GetCurrentNumberOfEntries, 168
hashMap_GetHashAtIndex, 168
hashMap_GetKeyAtIndex, 169
hashMap_GetMaxNumberOfEntries, 169
hashMap_GetPayload, 169
hashMap_GetULongPayload, 169
hashMap_Grow, 170
hashMap_IsOK, 170
hashMap_IsSorted, 170
hashMap_LoadToFile, 170
hashMap_SaveToFile, 170
hashMap_Sort, 170
hashmap_SwapRecords, 171
hashmap.h
hashFunction, 173
hashMap_Add, 173
hashMap_AddULong, 173
hashMap_Clear, 173
hashMap_ContainsKey, 173
hashMap_ContainsValue, 175
hashMap_Create, 175
hashMap_Destroy, 175

hashMap_FindIndex, 175
hashMap_GetCurrentNumberOfEntries, 176
hashMap_GetHashAtIndex, 176
hashMap_GetKeyAtIndex, 176
hashMap_GetMaxNumberOfEntries, 176
hashMap_GetPayload, 177
hashMap_GetULongPayload, 177
hashMap_LoadToFile, 177
hashMap_SaveToFile, 177
hashMap_Sort, 178
hashmap_SwapRecords, 178

hashmap_SwapRecords
hashmap.c, 171
hashmap.h, 178

headerRAW
HTTPHeader, 28

headerRAWSize
HTTPHeader, 28

headerResponse
AmmServer_DynamicRequest, 16

height
Image, 30
image_region_type, 31
image_win_type, 31

helloworld
helloworld.c, 56

helloworld.c
close_dynamic_content, 55
helloworld, 56
helloworld_times_shown, 56
init_dynamic_content, 55
main, 55
prepare_helloworld_content_callback, 56
templates_root, 56
webserver_root, 56

helloworld_times_shown
helloworld.c, 56

hidden
board, 21

hits
hashMapEntry, 26

hm_addLock
hashMap, 25

hm_fileLock
hashMap, 25

host
HTTPHeader, 28

hostLength
HTTPHeader, 28

hour
timestamp, 49

htmlContent, 26
currentDataLength, 27
data, 27
totalDataLength, 27

http_tools.h
AUDIO, 257
EXECUTABLE, 258
FOLDER, 258
IMAGE, 257
NO_FILETYPE, 257
RESERVED_CTE_VALUE, 257
TEXT, 257
VIDEO, 258

httpHeader.h
HTTPHEADER_ACCEPT_ENCODING, 223
HTTPHEADER_AUTHORIZATION, 223
HTTPHEADER_CONNECTION, 223
HTTPHEADER_COOKIE, 223
HTTPHEADER_EMPTY, 223
HTTPHEADER_END_OF_ITEMS, 224
HTTPHEADER_HOST, 223
HTTPHEADER_IF_MODIFIED_SINCE, 223
HTTPHEADER_IF_NONE_MATCH, 223
HTTPHEADER_RANGE, 223
HTTPHEADER_REFERER, 224
HTTPHEADER_REFERRER, 223
HTTPHEADER_USER_AGENT, 224

http_header_analysis.c
AnalyzeHTTPHeader, 179
AnalyzeHTTPLineRequest, 180
AppendPOSTRequestToHTTPHeader, 180
CR, 179
FreeHTTPHeader, 180
HTTPHeaderComplete, 180
HTTPHeaderIsPOST, 180
LF, 179
ProcessAuthorizationHTTPLine, 181
ProcessFirstHTTPLine, 181
ProcessRangeHTTPLine, 181
ReceiveHTTPHeader, 181

http_header_analysis.h
AnalyzeHTTPHeader, 182
AppendPOSTRequestToHTTPHeader, 182
FreeHTTPHeader, 183
HTTPHeaderComplete, 183
HTTPHeaderIsPOST, 183
ReceiveHTTPHeader, 183

http_tools.c
CheckHTTPHeaderCategory, 250
CheckHTTPHeaderCategoryAllCaps, 250
CheckIfFileIsVideo, 250
convertToUpperCase, 251
DirectoryExistsAmmServ, 251
encodeToBase64, 251
FileExistsAmmServ, 251
FilenameStripperOk, 251
FindIndexFile, 252
findOutClientIDOfPeer, 252
freeString, 252
GetContentType, 252
GetContentTypeForExtension, 252
GetExtensionImage, 252
GetExtentionType, 254
GetIntFromHTTPHeaderFieldPayload, 254
GetNewStringFromHTTPHeaderFieldPayload, 254

ReducePathSlashes_Inplace, 254
RequestHTTPWebPage, 254
seek_blank_char, 255
seek_non_blank_char, 255
ServerThreads_DropRootUID, 255
setSocketTimeouts, 255
strToUpcase, 256
StripGETRequestQueryAndFragment, 255
StripHTMLCharacters_Inplace, 255
StripVariableFromGETorPOSTString, 255
stristr, 256
stristr2Caps, 256
trim_last_empty_chars, 256
http_tools.h
CheckHTTPHeaderCategory, 258
CheckHTTPHeaderCategoryAllCaps, 258
CheckIfFileIsVideo, 258
contentType, 257
contentTypeEnumerator, 257
DirectoryExistsAmmServ, 258
encodeToBase64, 258
FileExistsAmmServ, 258
FilenameStripperOk, 259
FindIndexFile, 259
findOutClientIDOfPeer, 259
freeString, 259
GetContentType, 259
GetExtensionImage, 260
GetExtentionType, 260
GetIntFromHTTPHeaderFieldPayload, 260
GetNewStringFromHTTPHeaderFieldPayload, 260
ReducePathSlashes_Inplace, 260
RequestHTTPWebPage, 261
seek_blank_char, 261
seek_non_blank_char, 261
ServerThreads_DropRootUID, 261
setSocketTimeouts, 261
strToUpcase, 262
StripGETRequestQueryAndFragment, 261
StripHTMLCharacters_Inplace, 261
StripVariableFromGETorPOSTString, 262
trim_last_empty_chars, 262
httpHeader.c
scanFor_httpHeader, 222
httpHeader.h
scanFor_httpHeader, 224

i
Services/MyRemoteDesktop/xwd-1.0.5/main.c, 99
IMAGE
http_tools.h, 257
IMAGEFILES_BMP
imageFiles.h, 225
IMAGEFILES_DIB
imageFiles.h, 225
IMAGEFILES_EMPTY
imageFiles.h, 225
IMAGEFILES_END_OF_ITEMS
imageFiles.h, 225
IMAGEFILES_GIF
imageFiles.h, 225
IMAGEFILES_ICO
imageFiles.h, 225
IMAGEFILES_J2C
imageFiles.h, 225
IMAGEFILES_JPEG
imageFiles.h, 225
IMAGEFILES_JPG
imageFiles.h, 225
IMAGEFILES_PNG
imageFiles.h, 225
IMAGEFILES_PNM
imageFiles.h, 225
IMAGEFILES_PPM
imageFiles.h, 225
IMAGEFILES_RAW
imageFiles.h, 225
IMAGEFILES_RLE
imageFiles.h, 225
IMAGEFILES_SVG
imageFiles.h, 225
IMAGEFILES_TIFF
imageFiles.h, 225
IMAGEFILES_WEBP
imageFiles.h, 225
i_adapt
PassToPreSpawnedThread, 40
INIT_NAME
dsimple.h, 285
Image, 30
depth, 30
height, 30
imageSize, 30
pixels, 30
width, 30
imageFiles.h
IMAGEFILES_BMP, 225
IMAGEFILES_DIB, 225
IMAGEFILES_EMPTY, 225
IMAGEFILES_END_OF_ITEMS, 225
IMAGEFILES_GIF, 225
IMAGEFILES_ICO, 225
IMAGEFILES_J2C, 225
IMAGEFILES_JPEG, 225
IMAGEFILES_JPG, 225
IMAGEFILES_PNG, 225
IMAGEFILES_PNM, 225
IMAGEFILES_PPM, 225
IMAGEFILES_RAW, 225
IMAGEFILES_RLE, 225
IMAGEFILES_SVG, 225
IMAGEFILES_TIFF, 225
IMAGEFILES_WEBP, 225
Image_Size
Services/MyRemoteDesktop/xwd-1.0.5/main.c, 98
xwd.c, 298
image_region_type, 30

border, 31
cmap, 31
height, 31
vis, 31
visible_region, 31
width, 31
win, 31
x_rootrel, 31
x_vis, 31
y_rootrel, 31
y_vis, 31
image_win_type, 31
border_width, 31
cmap, 31
height, 31
parent, 31
vis, 31
width, 32
win, 32
x_rootrel, 32
x_vis, 32
y_rootrel, 32
y_vis, 32
imageFiles.c
scanFor_imageFiles, 224
imageFiles.h
scanFor_imageFiles, 226
imageSize
Image, 30
imageUID
board, 21
imaging.c
bitBltImage, 111
bitBltImageRotated, 111
copyImage, 111
createImage, 111
DISPLAY_DEBUG_INFO, 111
destroyImage, 111
PPMREADBUFLEN, 111
ReadPPM, 111
WritePPM, 111
imaging.h
bitBltImage, 112
copyImage, 112
createImage, 112
destroyImage, 112
ReadPPM, 112
WritePPM, 112
img_warp.c
ABS, 112
ABSDIFF, 112
coolPHPWave, 112
warpImage, 112
img_warp.h
coolPHPWave, 113
warpImage, 113
incomingHeader
HTTPTransaction, 29
index.c
destroy_index_prototype, 279
getFooterLinksHTML, 279
getLeftBlogRollHTML, 279
getMenuListHTML, 279
getPostListHTML, 279
getRightBlogRollHTML, 279
getWidgetListHTML, 279
indexPage, 279
loadPosts, 279
prepare_index, 279
prepare_index_prototype, 279
setupMyBlog, 279
strlimcpy, 279
index.h
destroy_index_prototype, 279
prepare_index, 280
indexContext
Services/MyTube/main.c, 101
indexPage
index.c, 279
main.cpp, 304
ScriptRunner/main.c, 81
Services/CinemaPilot/main.c, 88
Services/GeoPosShare/main.c, 91
Services/MyRemoteDesktop/main.c, 97
Services/MyTube/main.c, 101
Services/MyURL/main.c, 106
indexPageContext
Services/MyRemoteDesktop/main.c, 97
indexPageLength
Services/MyRemoteDesktop/main.c, 97
Services/MyURL/main.c, 106
indexPagePath
Services/MyRemoteDesktop/main.c, 97
Services/MyURL/main.c, 106
indexer.c
clearExtensionFAST, 299
getAVideoForQuery, 299
loadVideoDatabase, 299
path_cat2, 299
videoDefaultTestTranmission, 299
indexer.h
loadVideoDatabase, 300
MAX_STR, 300
path_cat2, 300
videoDefaultTestTranmission, 300
init_buffer
jpgInput.c, 114
init_dynamic_content
helloworld.c, 55
main.cpp, 303
ScriptRunner/main.c, 80
Services/AmmarServer/main.c, 83
Services/CinemaPilot/main.c, 87
Services/GeoPosShare/main.c, 90
Services/HabChan/main.c, 92
Services/MyBlog/main.c, 93

Services/MyLoader/main.c, 94
Services/MyRemoteDesktop/main.c, 96
Services/MyTube/main.c, 100
Services/MyURL/main.c, 104
Services/SimpleTemplate/main.c, 107
Services/SQLiteServer/main.c, 108
initFakeVisual
multiVis.c, 294
multiVis.h, 295
initXwdLib
Services/MyRemoteDesktop/xwd-1.0.5/main.c, 98
XwdLib.h, 298
InputParser, 32
∼InputParser, 32
DefaultDelimeterSetup, 33
GetDelimeter, 33
GetLowercaseWord, 33
GetUpcaseWord, 33
GetWord, 33
GetWordChar, 33
GetWordInt, 33
GetWordLength, 33
InputParser, 32
InputParser, 32
SeperateWords, 33
SeperateWordsCC, 33
SeperateWordsUC, 33
SetDelimeter, 33
Version, 33
InputParser.cpp
ver, 186
Version, 186
InputParser_C.c
_ipc_ver, 189
CheckDelimeterNumOk, 187
CheckIPCOk, 187
CheckWordNumOk, 187
InputParser_ClearNonCharacters, 187
InputParser_Create, 187
InputParser_DefaultDelimeters, 187
InputParser_Destroy, 188
InputParser_GetDelimeter, 188
InputParser_GetLowercaseWord, 188
InputParser_GetUpcaseWord, 188
InputParser_GetWord, 188
InputParser_GetWordChar, 188
InputParser_GetWordFloat, 188
InputParser_GetWordInt, 188
InputParser_GetWordLength, 188
InputParser_SelfCheck, 188
InputParser_SeperateWords, 188
InputParser_SeperateWordsCC, 188
InputParser_SeperateWordsUC, 189
InputParser_SetDelimeter, 189
InputParser_TrimCharacters, 189
InputParser_TrimCharactersEnd, 189
InputParser_TrimCharactersStart, 189
InputParser_WordCompare, 189
InputParser_WordCompareAuto, 189
InputParser_WordCompareNoCase, 189
InputParser_WordCompareNoCaseAuto, 189
InputParserC_Version, 189
Str2Int_internal, 189
warningsAboutIncorrectlyAllocatedStackIssued, 189
InputParser_C.h
CONTAINERS_MAX, 191
CheckWordNumOk, 191
DELIM_MAX_MAX, 191
InputParser_ClearNonCharacters, 191
InputParser_Create, 191
InputParser_DefaultDelimeters, 191
InputParser_Destroy, 191
InputParser_GetDelimeter, 191
InputParser_GetLowercaseWord, 191
InputParser_GetUpcaseWord, 191
InputParser_GetWord, 191
InputParser_GetWordChar, 192
InputParser_GetWordFloat, 192
InputParser_GetWordInt, 192
InputParser_GetWordLength, 192
InputParser_SelfCheck, 192
InputParser_SeperateWords, 192
InputParser_SeperateWordsCC, 192
InputParser_SeperateWordsUC, 192
InputParser_SetDelimeter, 192
InputParser_TrimCharacters, 192
InputParser_TrimCharactersEnd, 192
InputParser_TrimCharactersStart, 193
InputParser_WordCompare, 193
InputParser_WordCompareAuto, 193
InputParser_WordCompareNoCase, 193
InputParser_WordCompareNoCaseAuto, 193
InputParserC_Version, 193
MAX_COMPLICITY, 191
MAX_MEMORY, 191
MAX_STRING, 191
USE_SCANF, 191
InputParser_ClearNonCharacters
InputParser_C.c, 187
InputParser_C.h, 191
InputParser_Create
InputParser_C.c, 187
InputParser_C.h, 191
InputParser_DefaultDelimeters
InputParser_C.c, 187
InputParser_C.h, 191
InputParser_Destroy
InputParser_C.c, 188
InputParser_C.h, 191
InputParser_GetDelimeter
InputParser_C.c, 188
InputParser_C.h, 191
InputParser_GetLowercaseWord
InputParser_C.c, 188
InputParser_C.h, 191

InputParser_GetUpcaseWord
InputParser_C.c, 188
InputParser_C.h, 191
InputParser_GetWord
InputParser_C.c, 188
InputParser_C.h, 191
InputParser_GetWordChar
InputParser_C.c, 188
InputParser_C.h, 192
InputParser_GetWordFloat
InputParser_C.c, 188
InputParser_C.h, 192
InputParser_GetWordInt
InputParser_C.c, 188
InputParser_C.h, 192
InputParser_GetWordLength
InputParser_C.c, 188
InputParser_C.h, 192
InputParser_SelfCheck
InputParser_C.c, 188
InputParser_C.h, 192
InputParser_SeperateWords
InputParser_C.c, 188
InputParser_C.h, 192
InputParser_SeperateWordsCC
InputParser_C.c, 188
InputParser_C.h, 192
InputParser_SeperateWordsUC
InputParser_C.c, 189
InputParser_C.h, 192
InputParser_SetDelimeter
InputParser_C.c, 189
InputParser_C.h, 192
InputParser_TrimCharacters
InputParser_C.c, 189
InputParser_C.h, 192
InputParser_TrimCharactersEnd
InputParser_C.c, 189
InputParser_C.h, 192
InputParser_TrimCharactersStart
InputParser_C.c, 189
InputParser_C.h, 193
InputParser_WordCompare
InputParser_C.c, 189
InputParser_C.h, 193
InputParser_WordCompareAuto
InputParser_C.c, 189
InputParser_C.h, 193
InputParser_WordCompareNoCase
InputParser_C.c, 189
InputParser_C.h, 193
InputParser_WordCompareNoCaseAuto
InputParser_C.c, 189
InputParser_C.h, 193
InputParserC, 34
container_end, 34
container_start, 34
cur_container_count, 34
cur_delimeter_count, 34
delimeters, 34
guardbyte1, 34
guardbyte2, 34
guardbyte3, 34
guardbyte4, 34
local_allocation, 34
max_container_count, 35
max_delimeter_count, 35
str, 35
str_length, 35
tokenlist, 35
tokens_count, 35
tokens_max, 35
InputParserC_Version
InputParser_C.c, 189
InputParser_C.h, 193
instance
HTTPTransaction, 29
PassToHTTPThread, 39
PassToPreSpawnedThread, 40
PreSpawnedThread, 44
instance_CountFreeOP
server_configuration.c, 202
server_configuration.h, 212
instance_CountNewMallocOP
server_configuration.c, 202
server_configuration.h, 212
instance_WeCanCommitMoreMemory
server_configuration.c, 202
server_configuration.h, 212
instanceName
AmmServer_Instance, 18
interactContext
Services/MyTube/main.c, 101
interestPoints
Services/GeoPosShare/main.c, 91
IntermediateTests
AmmServerlib/InputParser/InputParser_C_Tester/main.c, 60
intermission
Services/CinemaPilot/main.c, 87
ip
PassToHTTPThread, 40
is_an_unsafe_str
Services/MyURL/main.c, 104
isURLDBSorted
Services/MyURL/main.c, 104
issueCommandToMplayer
Services/CinemaPilot/main.c, 87
item
_list_item, 15
linkItemList, 35
menuItemList, 36
playlist, 41
postItemList, 43
tagItemList, 47
widgetItemList, 54

joystickExecute
main.cpp, 303
ScriptRunner/main.c, 80
jpegtest
jpgInput.c, 114
jpgInput.c
empty_buffer, 114
fastJPGHeaderCheck, 114
init_buffer, 114
jpegtest, 114
ReadJPEG, 114
term_buffer, 114
WriteJPEGFile, 114
WriteJPEGInternal, 114
WriteJPEGMemory, 115
jpgInput.h
ReadJPEG, 115
USE_JPG_FILES, 115
WriteJPEGFile, 115
WriteJPEGMemory, 115

keep_var_on_stack
PassToHTTPThread, 40
keepalive
HTTPHeader, 28
keepalivePlaylist
Services/CinemaPilot/main.c, 87
key
hashMapEntry, 26
keyHash
hashMapEntry, 26
keyLength
hashMapEntry, 26

LESS
list.h, 289
LF
http_header_analysis.c, 179
LINE_MAX_LENGTH
state.h, 273
label
linkLabelItem, 36
widgetItem, 53
last_callback
AmmServer_RH_Context, 21
lastAuthenticationToken
UserAccountDatabase, 51
lastReply
thread, 47
layer
OverlayInfo, 38
OverlayVisualPropertyRec, 39
length
tokens, 49
likes
videoItem, 52
link
linkLabelItem, 36
widgetItem, 53
linkItemList, 35
currentItems, 35
item, 35
maxItems, 35
linkLabelItem, 35
label, 36
link, 36
links
Services/MyURL/main.c, 106
linksLeft
website, 53
linksRight
website, 53
list
list.h, 290
list.c
add_to_list, 287
delete_from_list, 287
delete_list, 287
delete_list_destroying, 287
dup_list_head, 287
first_in_list, 287
list_is_empty, 288
list_length, 288
new_list, 288
next_in_list, 288
zero_list, 288
list.h
add_to_list, 290
DESTRUCT_FUNC_PTR, 290
DUP_WHOLE_LIST, 289
delete_from_list, 290
delete_list, 290
delete_list_destroying, 290
dup_list_head, 290
EQUAL, 289
first_in_list, 291
GREATER, 289
LESS, 289
list, 290
list_is_empty, 291
list_item, 290
list_length, 291
list_ptr, 290
new_list, 291
next_in_list, 291
START_AT_CURR, 290
zero_list, 291
list_is_empty
list.c, 288
list.h, 291
list_item
list.h, 290
list_length
list.c, 288
list.h, 291
list_ptr
list.h, 290

loadBoardSettings
board.c, 269
LoadConfigurationFile
server_configuration.c, 203
server_configuration.h, 213
LoadMyURLDBFile
Services/MyURL/main.c, 104
loadPosts
index.c, 279
loadPostsFromSQL
database.c, 276
database.h, 278
loadSite
state.c, 271
state.h, 274
loadThread
thread.c, 275
loadVideoDatabase
indexer.c, 299
indexer.h, 300
loaded_cache_items
AmmServer_Instance, 18
loaded_cache_items_Kbytes
AmmServer_Instance, 18
loaded_links
Services/MyURL/main.c, 106
local_allocation
InputParserC, 34
logEcho
logs.h, 264
Services/AmmarServer/main.c, 83
logs.c
AccessLogAppend, 262
error, 262
ErrorLogAppend, 263
warning, 263
logs.h
AccessLogAppend, 264
BLACK, 264
BLUE, 264
BOLDBLACK, 264
BOLDBLUE, 264
BOLDCYAN, 264
BOLDGREEN, 264
BOLDMAGENTA, 264
BOLDRED, 264
BOLDWHITE, 264
BOLDYELLOW, 264
CYAN, 264
error, 264
ErrorLogAppend, 265
GREEN, 264
logEcho, 264
MAGENTA, 264
NORMAL, 264
RED, 264
WHITE, 264
warning, 265
YELLOW, 264
longURL
URLDB, 50
longestStringLength
fastStringParser, 24
lowbit
Services/MyRemoteDesktop/xwd-1.0.5/main.c, 98
xwd.c, 298

MAGENTA
AmmServerlib/InputParser/InputParser_C_-Tester/main.c, 60
logs.h, 264
testHashMap.c, 311
MAX
dsimple.h, 285
multiVis.c, 293
MAX_BINDING_PORT
main.cpp, 302
ScriptRunner/main.c, 80
Services/AmmarServer/main.c, 83
Services/GeoPosShare/main.c, 89
Services/HabChan/main.c, 92
Services/MyURL/main.c, 103
MAX_BOARDS
state.h, 273
MAX_CLIENT_THREADS
server_configuration.h, 209
MAX_COMMAND_SIZE
main.cpp, 302
ScriptRunner/main.c, 80
MAX_COMPLICITY
InputParser_C.h, 191
MAX_CONTENT
database.h, 277
MAX_CONTENT_TYPE
server_configuration.h, 210
MAX_ETAG_SIZE
server_configuration.h, 210
MAX_FILE_PATH
AmmServerlib.h, 120
MAX_IP_STRING_SIZE
AmmServerlib.h, 120
MAX_LINKS
Services/MyURL/main.c, 104
MAX_MEMORY
InputParser_C.h, 191
MAX_MENU_ITEMS
database.h, 277
MAX_QUERY
AmmServerlib.h, 120
MAX_RESOURCE
AmmServerlib.h, 120
MAX_STR
database.h, 277
indexer.h, 300
MAX_STRING
InputParser_C.h, 191
MAX_STRING_SIZE

state.h, 273
MAX_TAGS_PER_POST
database.h, 277
MAX_TO_SIZE
Services/MyURL/main.c, 104
MAX_WIDGET_ITEMS
database.h, 278
MAX_numberOfVideos
videoCollection, 51
MAXIMUM_LEVELS
fastStringParser.c, 307
MAXcompressedContentSize
AmmServer_DynamicRequest, 16
MAXcontentSize
AmmServer_DynamicRequest, 16
MAXstringsLoaded
fastStringParser, 24
MIN
dsimple.h, 285
multiVis.c, 293
main
AmmCaptcha/AmmCaptchaTester/main.c, 57
AmmServerlib/InputParser/InputParser_C_-Tester/main.c, 60
helloworld.c, 55
main.cpp, 303
myblogTool.c, 280
ScriptRunner/main.c, 80
Services/AmmarServer/main.c, 83
Services/CinemaPilot/main.c, 87
Services/GeoPosShare/main.c, 90
Services/HabChan/main.c, 92
Services/MyBlog/main.c, 93
Services/MyLoader/main.c, 95
Services/MyRemoteDesktop/main.c, 96
Services/MyRemoteDesktop/xwd-1.0.5/main.c, 98
Services/MyTube/main.c, 100
Services/MyURL/main.c, 105
Services/SimpleTemplate/main.c, 107
Services/SQLiteServer/main.c, 108
StringRecognizer/main.c, 109
testHashMap.c, 311
xwd.c, 298
main.cpp
ADMIN_BINDING_PORT, 302
admin_root, 304
admin_server, 304
base_image, 304
chatbox, 304
close_dynamic_content, 303
default_server, 304
ENABLE_ADMIN_PAGE, 302
ENABLE_CHAT_BOX, 302
EraseFile, 303
execute, 303
FileExistsTest, 303
form, 304
GET_override, 304
getBackCommandLine, 303
indexPage, 304
init_dynamic_content, 303
joystickExecute, 303
MAX_BINDING_PORT, 302
MAX_COMMAND_SIZE, 302
main, 303
page, 304
pageLength, 304
prepare_base_image, 303
prepare_form_content_callback, 303
prepare_index_content_callback, 303
prepare_stats_content_callback, 303
prepare_top_image, 303
random_chars, 304
replaceChar, 303
settings, 304
stats, 304
store_new_configuration_callback, 304
StringIsHTMLSafe, 304
templates_root, 304
termination_handler, 304
top_image, 304
webserver_root, 304
MainHTTPServerThread
threadedServer.c, 241
mallocHTMLListOfThreadsOfBoard
thread.c, 275
max_container_count
InputParserC, 35
max_delimeter_count
InputParserC, 35
max_ret_word
AmmServerlib/InputParser/InputParser_C_-Tester/main.c, 60
maxItems
linkItemList, 35
menuItemList, 36
playlist, 41
widgetItemList, 54
maxNumberOfBoards
site, 45
maxNumberOfEntries
hashMap, 25
maxNumberOfReplies
thread, 47
maxPosts
postItemList, 43
maxTags
tagItemList, 47
maxThreads
board, 22
menu
website, 53
menuItemList, 36
currentItems, 36
item, 36
maxItems, 36

message
    post, 42
messageSize
    post, 42
minute
    timestamp, 49
modification
    cache_item, 23
month
    timestamp, 49
months
    time_provider.c, 266
movieList
    Services/CinemaPilot/main.c, 88
mplayerControllerPath
    Services/CinemaPilot/main.c, 88
multiVis.c
    BLUE_SHIFT, 293
    DIRECT_COLOR, 293
    FreeXVisualInfo, 294
    GRAY_SCALE, 293
    GREEN_SHIFT, 293
    GetMultiVisualRegions, 294
    GetXVisualInfo, 294
    initFakeVisual, 294
    MAX, 293
    MIN, 293
    myBOX, 293
    myBoxPtr, 293
    myBoxRec, 294
    myREGION, 294
    PSEUDO_COLOR, 293
    RED_SHIFT, 293
    ReadAreaToImage, 294
    SAME_REGIONS, 293
    STATIC_GRAY, 293
    TRUE_COLOR, 293
multiVis.h
    GetMultiVisualRegions, 295
    initFakeVisual, 295
    ReadAreaToImage, 295
my_XRegion, 36
    extents, 36
    numRects, 36
    rects, 36
    size, 37
myBOX
    multiVis.c, 293
myBox, 37
    x1, 37
    x2, 37
    y1, 37
    y2, 37
myBoxPtr
    multiVis.c, 293
myBoxRec
    multiVis.c, 294
myREGION
    multiVis.c, 294
myStupidMemcpy
    AString.c, 136
myTube
    Services/MyTube/main.c, 101
myblog
    database.c, 276
    database.h, 278
myblogTool.c
    main, 280
    SQL_appendpost, 280
    SQL_close, 280
    SQL_error, 280
    SQL_getVersion, 280
    SQL_init, 280
    sqlserver, 280
myurl_server
    Services/MyURL/main.c, 106

NO_FILETYPE
    http_tools.h, 257
NONE
    AmmServerlib.h, 121
NUMBER_OF_COMMANDS
    Services/CinemaPilot/main.c, 86
NUMBER_OF_FILETYPES
    state.h, 273
NUMBER_OF_STATES
    Services/CinemaPilot/main.c, 86
NORMAL
    AmmServerlib/InputParser/InputParser_C_Tester/main.c, 60
    AString.c, 135
    logs.h, 264
    testHashMap.c, 311
NOT_FLEXIBLE
    wsutils.h, 296
NXTOPT
    dsimple.c, 283
NXTOPTP
    dsimple.c, 283
name
    board, 22
new_list
    list.c, 288
    list.h, 291
next
    _list_item, 15
next_in_list
    list.c, 288
    list.h, 291
None
    wsutils.h, 296
numRects
    my_XRegion, 36
numberOfBoards
    site, 45
numberOfComplaints
    post, 42

numberOfImages
thread, 47
numberOfItems
playlist, 41
numberOfLoadedVideos
videoCollection, 51
numberOfReplies
thread, 47

OPTIONS
AmmServerlib.h, 121
OPTION
dsimple.c, 283
op
post, 42
thread, 47
Open_Display
dsimple.c, 284
dsimple.h, 286
ourSite
state.c, 272
state.h, 274
out_file
Services/MyRemoteDesktop/xwd-1.0.5/main.c, 99
outgoingBody
HTTPTransaction, 29
outgoingBodySize
HTTPTransaction, 29
outl
dsimple.c, 284
dsimple.h, 286
OverlayInfo, 38
layer, 38
pOverlayVisualInfo, 38
transparentType, 38
value, 38
OverlayVisualPropertyRec, 38
layer, 39
transparentType, 39
value, 39
visualID, 39

PATCH
AmmServerlib.h, 121
POST
AmmServerlib.h, 121
POSTHEADER_CONTENT_DISPOSITION
postHeader.h, 228
POSTHEADER_CONTENT_LENGTH
postHeader.h, 228
POSTHEADER_CONTENT_TYPE
postHeader.h, 228
POSTHEADER_EMPTY
postHeader.h, 228
POSTHEADER_END_OF_ITEMS
postHeader.h, 228
PUT
AmmServerlib.h, 121
PACKAGE
config.h, 282
PACKAGE_BUGREPORT
config.h, 282
PACKAGE_NAME
config.h, 282
PACKAGE_STRING
config.h, 282
PACKAGE_TARNAME
config.h, 282
PACKAGE_URL
config.h, 282
PACKAGE_VERSION
config.h, 282
PASSWORD
AmmServer_Instance_Settings, 19
POPEN_BUFFER_SIZE
AmmServerlib.h, 120
POST_request
AmmServer_DynamicRequest, 16
POST_request_length
AmmServer_DynamicRequest, 16
POSTrequest
HTTPHeader, 28
POSTrequestSize
HTTPHeader, 28
pOverlayVisualInfo
OverlayInfo, 38
PPMREADBUFLEN
imaging.c, 111
PSEUDO_COLOR
multiVis.c, 293
page
main.cpp, 304
ScriptRunner/main.c, 81
pageLength
main.cpp, 304
ScriptRunner/main.c, 81
parent
image_win_type, 31
ParseString
AmmServerlib/InputParser/InputParser_C_-Tester/main.c, 60
PassToHTTPThread, 39
client, 39
clientlen, 39
clientsock, 39
instance, 39
ip, 40
keep_var_on_stack, 40
port, 40
pre_spawned_thread, 40
thread_id, 40
PassToPreSpawnedThread, 40
i_adapt, 40
instance, 40
password
post, 42
thread, 47

path_cat
    directory_lists.c, 247
path_cat2
    indexer.c, 299
    indexer.h, 300
pause_server
    AmmServer_Instance, 18
pauseMplayer
    Services/CinemaPilot/main.c, 87
payload
    hashMapEntry, 26
payloadLength
    hashMapEntry, 26
Pixel
    Services/MyRemoteDesktop/xwd-1.0.5/main.c, 98
    xwd.c, 298
pixels
    Image, 30
playFile
    playlistItem, 41
playlist, 40
    item, 41
    maxItems, 41
    numberOfItems, 41
    playlistActiveItem, 41
    playlistState, 41
playlistActiveItem
    playlist, 41
playlistItem, 41
    command, 41
    playFile, 41
    stopTime, 41
    triggerTime, 41
playlistState
    playlist, 41
port
    PassToHTTPThread, 40
post, 41
    creation, 42
    fileCachedName, 42
    fileDimensionHeight, 42
    fileDimensionWidth, 42
    fileOriginalName, 42
    fileType, 42
    hasFile, 42
    message, 42
    messageSize, 42
    numberOfComplaints, 42
    op, 42
    password, 42
    website, 53
postHeader.h
    POSTHEADER_CONTENT_DISPOSITION, 228
    POSTHEADER_CONTENT_LENGTH, 228
    POSTHEADER_CONTENT_TYPE, 228
    POSTHEADER_EMPTY, 228
    POSTHEADER_END_OF_ITEMS, 228
post_header_analysis.c
    AnalyzePOSTLineRequest, 184
post_header_analysis.h
    AnalyzePOSTLineRequest, 185
postHeader.c
    scanFor_postHeader, 227
postHeader.h
    scanFor_postHeader, 228
postItem, 42
    author, 43
    content, 43
    dateStr, 43
    tags, 43
    title, 43
postItemList, 43
    currentPosts, 43
    item, 43
    maxPosts, 43
postReceiver
    Services/HabChan/main.c, 92
postReceiver.c
    processPostReceiver, 270
postReceiver.h
    processPostReceiver, 270
postUID
    board, 22
pre_spawned_thread
    PassToHTTPThread, 40
PreSpawnThreads
    prespawnedThreads.c, 237
    prespawnedThreads.h, 240
PreSpawnedThread, 43
    busy, 44
    client, 44
    clientlen, 44
    clientsock, 44
    instance, 44
    prespawnedThreads.c, 237
    templates_root, 44
    thread_id, 44
    threadNum, 44
    webserver_root, 44
prepare_apk_link
    Services/GeoPosShare/main.c, 90
prepare_base_image
    main.cpp, 303
    ScriptRunner/main.c, 80
prepare_cars_content_callback
    Services/SQLiteServer/main.c, 109
prepare_chatbox_content_callback
    Services/AmmarServer/main.c, 83
prepare_command_content_callback
    Services/MyRemoteDesktop/main.c, 96
prepare_form_content_callback
    main.cpp, 303
    ScriptRunner/main.c, 80
    Services/AmmarServer/main.c, 84
prepare_gps_content_callback
    Services/AmmarServer/main.c, 84

Services/GeoPosShare/main.c, 90
prepare_helloworld_content_callback
helloworld.c, 56
prepare_index
index.c, 279
index.h, 280
prepare_index_content_callback
main.cpp, 303
ScriptRunner/main.c, 80
Services/MyRemoteDesktop/main.c, 96
prepare_index_prototype
index.c, 279
prepare_indexPage
Services/CinemaPilot/main.c, 87
Services/GeoPosShare/main.c, 90
prepare_interestPoints
Services/GeoPosShare/main.c, 90
prepare_random_content_callback
Services/AmmarServer/main.c, 84
Services/CinemaPilot/main.c, 87
Services/MyBlog/main.c, 93
Services/SimpleTemplate/main.c, 107
prepare_remoteControl_callback
Services/CinemaPilot/main.c, 87
prepare_screen_content_callback
Services/MyRemoteDesktop/main.c, 97
prepare_stats_content_callback
main.cpp, 303
ScriptRunner/main.c, 81
Services/AmmarServer/main.c, 84
Services/CinemaPilot/main.c, 87
Services/MyLoader/main.c, 95
Services/SimpleTemplate/main.c, 107
Services/SQLiteServer/main.c, 109
prepare_top_image
main.cpp, 303
ScriptRunner/main.c, 81
prepareBoardIndexView
board.c, 269
board.h, 269
prepareThreadIndexView
thread.c, 275
thread.h, 275
prepareThreadView
thread.c, 275
thread.h, 275
prespawn_jobs_finished
AmmServer_Instance, 18
prespawn_jobs_started
AmmServer_Instance, 18
prespawn_turn_to_serve
AmmServer_Instance, 18
prespawned_pool
AmmServer_Instance, 18
prespawnedThreadFlag
HTTPTransaction, 29
prespawnedThreads.c
PreSpawnThreads, 237
PreSpawnedThread, 237
UsePreSpawnedThreadToServeNewClient, 239
prespawnedThreads.h
PreSpawnThreads, 240
UsePreSpawnedThreadToServeNewClient, 240
printAllEnumeratorItems
fastStringParser.c, 308
printCars
sqlite.c, 305
printIfAllPossibleStrings
fastStringParser.c, 308
printURLDB
Services/MyURL/main.c, 105
ProcessAuthorizationHTTPLine
http_header_analysis.c, 181
processCommand
Services/CinemaPilot/main.c, 87
ProcessFirstHTTPLine
http_header_analysis.c, 181
processPostReceiver
postReceiver.c, 270
postReceiver.h, 270
ProcessRangeHTTPLine
http_header_analysis.c, 181
processUploadCallback
Services/MyLoader/main.c, 95
program_name
dsimple.c, 284
dsimple.h, 286
ptr
_list_item, 15

RESERVED_CTE_VALUE
http_tools.h, 257
RC_FILEVERSION
version.h, 268
RED
AmmServerlib/InputParser/InputParser_C_-Tester/main.c, 60
AString.c, 135
logs.h, 264
testHashMap.c, 311
RED_SHIFT
multiVis.c, 293
RH_Scenario
AmmServer_RH_Context, 21
RHScenarios
AmmServerlib.h, 121
random_chars
main.cpp, 304
ScriptRunner/main.c, 81
Services/AmmarServer/main.c, 84
Services/CinemaPilot/main.c, 88
Services/MyBlog/main.c, 94
Services/MyTube/main.c, 101
Services/SimpleTemplate/main.c, 107
Services/SQLiteServer/main.c, 109
randomVideoFileContext
Services/MyTube/main.c, 101

range_end
HTTPHeader, 28
range_start
HTTPHeader, 28
rc
SQLiteSession, 46
ReWriteMyURLDBFile
Services/MyURL/main.c, 105
ReadAreaToImage
multiVis.c, 294
multiVis.h, 295
ReadJPEG
jpgInput.c, 114
jpgInput.h, 115
ReadPPM
imaging.c, 111
imaging.h, 112
readPlaylist
Services/CinemaPilot/main.c, 87
ReceiveHTTPHeader
http_header_analysis.c, 181
http_header_analysis.h, 183
rects
my_XRegion, 36
recursiveTraverser
fastStringParser.c, 308
ReducePathSlashes_Inplace
http_tools.c, 254
http_tools.h, 260
referer
HTTPHeader, 28
refererLength
HTTPHeader, 28
remoteControl
Services/CinemaPilot/main.c, 88
RenderString
AmmCaptcha/main.c, 59
replaceChar
main.cpp, 303
ScriptRunner/main.c, 81
repliable
thread, 47
replies
thread, 47
request
AmmServer_RequestOverride_Context, 20
request_override_callback
AmmServer_RequestOverride_Context, 20
Services/AmmarServer/main.c, 84
Services/CinemaPilot/main.c, 87
Services/GeoPosShare/main.c, 90
Services/MyBlog/main.c, 93
Services/MyLoader/main.c, 95
Services/SimpleTemplate/main.c, 107
Services/SQLiteServer/main.c, 109
requestContext
AmmServer_RH_Context, 21
RequestHTTPWebPage
http_tools.c, 254
http_tools.h, 261
requestHeader
AmmServer_RequestOverride_Context, 20
requestResolver
Services/MyURL/main.c, 106
requestType
HTTPHeader, 28
res
SQLiteSession, 46
resolveRequest
Services/MyURL/main.c, 105
ResortDB
Services/MyURL/main.c, 105
resource
HTTPHeader, 29
resource_name
AmmServer_RH_Context, 21
resourceCacheID
HTTPTransaction, 30
resumeMplayer
Services/CinemaPilot/main.c, 88

SAME_PAGE_FOR_ALL_CLIENTS
AmmServerlib.h, 121
STATE_FINISHED
Services/CinemaPilot/main.c, 86
STATE_PLAYING
Services/CinemaPilot/main.c, 86
STATE_UNINITIALIZED
Services/CinemaPilot/main.c, 86
SAME_REGIONS
multiVis.c, 293
SB_CMAP_TYPE_FULL
wsutils.h, 296
SQL_appendpost
myblogTool.c, 280
SQL_close
database.c, 276
database.h, 278
myblogTool.c, 280
sqlite.c, 305
sqlite.h, 306
SQL_createInitialTables
database.c, 276
database.h, 278
SQL_error
database.c, 276
myblogTool.c, 280
SQL_getVersion
database.c, 276
myblogTool.c, 280
sqlite.c, 305
sqlite.h, 306
SQL_init
database.c, 276
database.h, 278
myblogTool.c, 280
sqlite.c, 305

sqlite.h, 306
SQL_populate
sqlite.c, 305
sqlite.h, 306
SQLiteSession, 45
db, 46
err_msg, 46
rc, 46
res, 46
START_AT_CURR
list.h, 290
STATIC_GRAY
multiVis.c, 293
STDC_HEADERS
config.h, 282
saveDynamicRequest
dynamic_requests.c, 144
dynamic_requests.h, 146
scanFor_applicationFiles
applicationFiles.c, 214
applicationFiles.h, 216
scanFor_archiveFiles
archiveFiles.c, 217
archiveFiles.h, 218
scanFor_audioFiles
audioFiles.c, 219
audioFiles.h, 220
scanFor_firstLines
firstLines.c, 221
firstLines.h, 222
scanFor_httpHeader
httpHeader.c, 222
httpHeader.h, 224
scanFor_imageFiles
imageFiles.c, 224
imageFiles.h, 226
scanFor_postHeader
postHeader.c, 227
postHeader.h, 228
scanFor_textFiles
textFiles.c, 229
textFiles.h, 230
scanFor_videoFiles
videoFiles.c, 230
videoFiles.h, 232
screen
dsimple.c, 284
dsimple.h, 286
screenContext
Services/MyRemoteDesktop/main.c, 97
ScriptRunner/main.c
ADMIN_BINDING_PORT, 80
admin_root, 81
admin_server, 81
base_image, 81
chatbox, 81
close_dynamic_content, 80
default_server, 81
ENABLE_ADMIN_PAGE, 80
ENABLE_CHAT_BOX, 80
execute, 80
form, 81
GET_override, 81
getBackCommandLine, 80
indexPage, 81
init_dynamic_content, 80
joystickExecute, 80
MAX_BINDING_PORT, 80
MAX_COMMAND_SIZE, 80
main, 80
page, 81
pageLength, 81
prepare_base_image, 80
prepare_form_content_callback, 80
prepare_index_content_callback, 80
prepare_stats_content_callback, 81
prepare_top_image, 81
random_chars, 81
replaceChar, 81
settings, 81
stats, 81
store_new_configuration_callback, 81
templates_root, 81
termination_handler, 81
top_image, 81
webserver_root, 82
second
timestamp, 49
seek_blank_char
http_tools.c, 255
http_tools.h, 261
seek_non_blank_char
http_tools.c, 255
http_tools.h, 261
Select_Window
dsimple.c, 284
dsimple.h, 286
Select_Window_Args
dsimple.c, 284
dsimple.h, 286
SendAuthorizationHeader
sendHTTPHeader.c, 197
sendHTTPHeader.h, 199
SendErrorCodeHeader
sendHTTPHeader.c, 198
sendHTTPHeader.h, 199
SendErrorFile
file_server.c, 194
file_server.h, 196
SendFile
file_server.c, 194
file_server.h, 196
sendHTTPHeader.c
SendAuthorizationHeader, 197
SendErrorCodeHeader, 198
SendNotModifiedHeader, 198

SendSuccessCodeHeader, 198
sendHTTPHeader.h
SendAuthorizationHeader, 199
SendErrorCodeHeader, 199
SendNotModifiedHeader, 200
SendSuccessCodeHeader, 200
SendMemoryBlockAsFile
file_server.c, 195
file_server.h, 197
SendNotModifiedHeader
sendHTTPHeader.c, 198
sendHTTPHeader.h, 200
SendPart
file_server.c, 195
SendSuccessCodeHeader
sendHTTPHeader.c, 198
sendHTTPHeader.h, 200
SeperateWords
InputParser, 33
SeperateWordsCC
InputParser, 33
SeperateWordsUC
InputParser, 33
serve_captcha_page
Services/MyURL/main.c, 105
serve_create_url_page
Services/MyURL/main.c, 105
serve_error_url_page
Services/MyURL/main.c, 105
serve_favicon
Services/MyTube/main.c, 100
serve_goto_url_page
Services/MyURL/main.c, 105
serve_index
Services/MyTube/main.c, 101
serve_interact
Services/MyTube/main.c, 101
serve_random_videopage
Services/MyTube/main.c, 101
serve_thumbnail
Services/MyTube/main.c, 101
serve_videofile
Services/MyTube/main.c, 101
serve_videopage
Services/MyTube/main.c, 101
serveCarsPageWithSQL
sqlite.c, 305
sqlite.h, 306
ServeClient
clientServer.c, 233
clientServer.h, 234
ServeClientKeepAliveLoop
clientServer.c, 234
server_configuration.c
AccessLog, 203
AccessLogEnable, 203
AssignStr, 201
CACHING_ENABLED, 203
CHANGE_PRIORITY, 203
CHANGE_TO_UID, 203
EmmitPossibleConfigurationWarnings, 201
ErrorLog, 203
ErrorLogEnable, 203
instance_CountFreeOP, 202
instance_CountNewMallocOP, 202
instance_WeCanCommitMoreMemory, 202
LoadConfigurationFile, 203
SetUsernameAndPassword, 203
TemplatesInternalURI, 204
varSocketTimeoutREAD_seconds, 204
varSocketTimeoutWRITE_seconds, 204
server_configuration.h
AccessLog, 213
AccessLogEnable, 213
AssignStr, 212
CACHING_ENABLED, 213
CHANGE_PRIORITY, 213
CHANGE_TO_UID, 213
ENABLE_POST, 209
EmmitPossibleConfigurationWarnings, 212
ErrorLog, 213
ErrorLogEnable, 213
instance_CountFreeOP, 212
instance_CountNewMallocOP, 212
instance_WeCanCommitMoreMemory, 212
LoadConfigurationFile, 213
MAX_CONTENT_TYPE, 210
MAX_ETAG_SIZE, 210
SetUsernameAndPassword, 213
TemplatesInternalURI, 214
varSocketTimeoutREAD_seconds, 214
varSocketTimeoutWRITE_seconds, 214
server_running
AmmServer_Instance, 18
server_thread_id
AmmServer_Instance, 18
ServerThreads_DropRootUID
http_tools.c, 255
http_tools.h, 261
serversock
AmmServer_Instance, 18
service_filename
Services/MyURL/main.c, 106
service_filename_noslash
Services/MyURL/main.c, 106
service_root
Services/MyURL/main.c, 106
service_root_withoutfilename
Services/MyURL/main.c, 106
Services/CinemaPilot/main.c
CMD_TYPE_BELL_OFF, 86
CMD_TYPE_BELL_ON, 86
CMD_TYPE_INTERMISSION, 86
CMD_TYPE_LIGHTS_OFF, 86
CMD_TYPE_LIGHTS_ON, 86
CMD_TYPE_MOVIE, 86

CMD_TYPE_NONE, 86
CMD_TYPE_SOUND_OFF, 86
CMD_TYPE_SOUND_ON, 86
CMD_TYPE_TRAILER, 86
NUMBER_OF_COMMANDS, 86
NUMBER_OF_STATES, 86
STATE_FINISHED, 86
STATE_PLAYING, 86
STATE_UNINITIALIZED, 86
Services/AmmarServer/main.c
admin_root, 84
admin_server, 84
chatbox, 84
close_dynamic_content, 83
debug_get_callback, 83
default_server, 84
ENABLE_CHAT_BOX, 83
ENABLE_STOP_PAGE, 83
executeScript, 84
executeScriptFunction, 83
executeScriptRC, 84
form, 84
fresh, 84
GET_override, 84
getdbg, 84
gps, 84
init_dynamic_content, 83
logEcho, 83
MAX_BINDING_PORT, 83
main, 83
prepare_chatbox_content_callback, 83
prepare_form_content_callback, 84
prepare_gps_content_callback, 84
prepare_random_content_callback, 84
prepare_stats_content_callback, 84
random_chars, 84
request_override_callback, 84
stats, 84
stop, 84
stop_callback, 84
templates_root, 85
WEBSERVERROOT, 83
webserver_root, 85
Services/CinemaPilot/main.c
close_dynamic_content, 87
commandType, 86
default_server, 88
executePlaylist, 87
executePlaylistCurrentItem, 87
fullScreenViewerPath, 88
GET_override, 88
indexPage, 88
init_dynamic_content, 87
intermission, 87
issueCommandToMplayer, 87
keepalivePlaylist, 87
main, 87
movieList, 88
mplayerControllerPath, 88
pauseMplayer, 87
prepare_indexPage, 87
prepare_random_content_callback, 87
prepare_remoteControl_callback, 87
prepare_stats_content_callback, 87
processCommand, 87
random_chars, 88
readPlaylist, 87
remoteControl, 88
request_override_callback, 87
resumeMplayer, 88
startMplayer, 88
stateType, 86
stats, 88
stopMplayer, 88
templates_root, 88
webserver_root, 88
Services/GeoPosShare/main.c
admin_root, 90
android, 90
apk, 90
appendGPS_OSM_Format, 90
appendGPSMessage, 90
close_dynamic_content, 90
default_server, 90
GET_override, 90
gps, 91
indexPage, 91
init_dynamic_content, 90
interestPoints, 91
main, 90
prepare_apk_link, 90
prepare_gps_content_callback, 90
prepare_indexPage, 90
prepare_interestPoints, 90
request_override_callback, 90
templates_root, 91
webserver_root, 91
Services/HabChan/main.c
boardIndexView, 92
close_dynamic_content, 92
init_dynamic_content, 92
MAX_BINDING_PORT, 92
main, 92
postReceiver, 92
templates_root, 92
threadIndexView, 92
threadView, 92
WEBSERVERROOT, 92
webserver_root, 92
Services/MyBlog/main.c
close_dynamic_content, 93
default_server, 93
GET_override, 93
init_dynamic_content, 93
main, 93
prepare_random_content_callback, 93

random_chars, 94
request_override_callback, 93
stats, 94
templates_root, 94
webserver_root, 94
Services/MyLoader/main.c
close_dynamic_content, 94
default_server, 95
GET_override, 95
init_dynamic_content, 94
main, 95
prepare_stats_content_callback, 95
processUploadCallback, 95
request_override_callback, 95
stats, 95
templates_root, 95
uploadProcessor, 95
webserver_root, 95
Services/MyRemoteDesktop/main.c
close_dynamic_content, 96
commandContext, 97
default_server, 97
GET_override, 97
indexPage, 97
indexPageContext, 97
indexPageLength, 97
indexPagePath, 97
init_dynamic_content, 96
main, 96
prepare_command_content_callback, 96
prepare_index_content_callback, 96
prepare_screen_content_callback, 97
screenContext, 97
templates_root, 97
webserver_root, 97
XWDLIB_BRIDGE, 96
Services/MyRemoteDesktop/xwd-1.0.5/main.c
_swaplong, 98
_swapshort, 98
closeXwdLib, 98
FEEP_VOLUME, 98
frame_only, 99
Get_XColors, 98
getScreen, 98
i, 99
Image_Size, 98
initXwdLib, 98
lowbit, 98
main, 98
out_file, 99
Pixel, 98
target_win, 99
usage, 98
Window_Dump, 99
Services/MyTube/main.c
close_dynamic_content, 100
database_root, 101
default_server, 101
favicon, 101
faviconContext, 101
GET_override, 101
indexContext, 101
indexPage, 101
init_dynamic_content, 100
interactContext, 101
main, 100
myTube, 101
random_chars, 101
randomVideoFileContext, 101
serve_favicon, 100
serve_index, 101
serve_interact, 101
serve_random_videopage, 101
serve_thumbnail, 101
serve_videofile, 101
serve_videopage, 101
templates_root, 101
thumbnailAllVideoDatabase, 101
thumbnailContext, 102
video_root, 102
videoFileContext, 102
videoPageContext, 102
webserver_root, 102
Services/MyURL/main.c
Add_MyURL, 104
allocateLinksIfNeeded, 104
allocated_links, 105
Append2MyURLDBFile, 104
captcha_url, 105
close_dynamic_content, 104
create_url, 105
db_addIDLock, 105
db_file, 105
db_fileLock, 105
default_failed, 106
error_url, 106
Find_longURL, 104
Find_longURLSerial, 104
Get_longURL, 104
goto_url, 106
hashURL, 104
indexPage, 106
indexPageLength, 106
indexPagePath, 106
init_dynamic_content, 104
is_an_unsafe_str, 104
isURLDBSorted, 104
links, 106
LoadMyURLDBFile, 104
loaded_links, 106
MAX_LINKS, 104
MAX_TO_SIZE, 104
main, 105
myurl_server, 106
printURLDB, 105
ReWriteMyURLDBFile, 105

requestResolver, 106
resolveRequest, 105
ResortDB, 105
serve_captcha_page, 105
serve_create_url_page, 105
serve_error_url_page, 105
serve_goto_url_page, 105
service_filename, 106
service_filename_noslash, 106
service_root, 106
service_root_withoutfilename, 106
sorted_links, 106
struct_cmp_urldb_items, 105
templates_root, 106
webserver_root, 106

Services/SQLiteServer/main.c
- close_dynamic_content, 108
- default_server, 109
- GET_override, 109
- init_dynamic_content, 108
- main, 108
- prepare_cars_content_callback, 109
- prepare_stats_content_callback, 109
- random_chars, 109
- request_override_callback, 109
- sqliteSession, 109
- stats, 109
- templates_root, 109
- webserver_root, 109

Services/SimpleTemplate/main.c
- close_dynamic_content, 107
- default_server, 107
- GET_override, 107
- init_dynamic_content, 107
- main, 107
- prepare_random_content_callback, 107
- prepare_stats_content_callback, 107
- random_chars, 107
- request_override_callback, 107
- stats, 107
- templates_root, 107
- webserver_root, 108

SetDelimeter
- InputParser, 33

setSocketTimeouts
- http_tools.c, 255
- http_tools.h, 261

SetUsernameAndPassword
- server_configuration.c, 203
- server_configuration.h, 213

settings
- AmmServer_Instance, 18
- main.cpp, 304
- ScriptRunner/main.c, 81

Setup_Display_And_Screen
- dsimple.c, 284
- dsimple.h, 286

Setup_Null_Display_And_Screen
- dsimple.c, 284
- dsimple.h, 286

setupMyBlog
- index.c, 279

shortURL
- URLDB, 50

shortURLHash
- URLDB, 50

shortestStringLength
- fastStringParser, 24

site, 44
- boards, 45
- maxNumberOfBoards, 45
- numberOfBoards, 45
- siteDescription, 45
- siteName, 45

siteDescription
- site, 45
- website, 53

siteName
- site, 45
- website, 53

siteURL
- website, 53

size
- my_XRegion, 37

social
- website, 53

socialLinks, 45
- facebookURL, 45
- twitterURL, 45
- youtubeURL, 45

sorted_links
- Services/MyURL/main.c, 106

SpawnThreadToServeNewClient
- freshThreads.c, 235
- freshThreads.h, 236

sqlite.c
- printCars, 305
- SQL_close, 305
- SQL_getVersion, 305
- SQL_init, 305
- SQL_populate, 305
- serveCarsPageWithSQL, 305

sqlite.h
- SQL_close, 306
- SQL_getVersion, 306
- SQL_init, 306
- SQL_populate, 306
- serveCarsPageWithSQL, 306

sqliteSession
- Services/SQLiteServer/main.c, 109

sqlserver
- database.c, 276
- database.h, 278
- myblogTool.c, 280

src/AmmCaptcha/AmmCaptcha.h, 56
src/AmmCaptcha/AmmCaptchaTester/main.c, 57

src/AmmCaptcha/imaging.c, 110
src/AmmCaptcha/imaging.h, 111
src/AmmCaptcha/img_warp.c, 112
src/AmmCaptcha/img_warp.h, 113
src/AmmCaptcha/jpgInput.c, 113
src/AmmCaptcha/jpgInput.h, 115
src/AmmCaptcha/main.c, 57
src/AmmServerlib/AString/AString.c, 134
src/AmmServerlib/AString/AString.h, 136
src/AmmServerlib/AmmServerlib.h, 115
src/AmmServerlib/InputParser/InputParser.cpp, 185
src/AmmServerlib/InputParser/InputParser.h, 186
src/AmmServerlib/InputParser/InputParser_C.c, 186
src/AmmServerlib/InputParser/InputParser_C.h, 190
src/AmmServerlib/InputParser/InputParser_C_Tester/main.c, 59
src/AmmServerlib/cache/client_list.c, 137
src/AmmServerlib/cache/client_list.h, 139
src/AmmServerlib/cache/dynamic_requests.c, 142
src/AmmServerlib/cache/dynamic_requests.h, 145
src/AmmServerlib/cache/file_caching.c, 147
src/AmmServerlib/cache/file_caching.h, 154
src/AmmServerlib/cache/file_compression.c, 162
src/AmmServerlib/cache/file_compression.h, 163
src/AmmServerlib/hashmap/hashmap.c, 165
src/AmmServerlib/hashmap/hashmap.h, 171
src/AmmServerlib/header_analysis/http_header_analysis.c, 178
src/AmmServerlib/header_analysis/http_header_analysis.h, 181
src/AmmServerlib/header_analysis/post_header_analysis.c, 184
src/AmmServerlib/header_analysis/post_header_analysis.h, 185
src/AmmServerlib/main.c, 60
src/AmmServerlib/network/file_server.c, 193
src/AmmServerlib/network/file_server.h, 195
src/AmmServerlib/network/sendHTTPHeader.c, 197
src/AmmServerlib/network/sendHTTPHeader.h, 199
src/AmmServerlib/server_configuration.c, 200
src/AmmServerlib/server_configuration.h, 204
src/AmmServerlib/stringscanners/applicationFiles.c, 214
src/AmmServerlib/stringscanners/applicationFiles.h, 215
src/AmmServerlib/stringscanners/archiveFiles.c, 217
src/AmmServerlib/stringscanners/archiveFiles.h, 217
src/AmmServerlib/stringscanners/audioFiles.c, 219
src/AmmServerlib/stringscanners/audioFiles.h, 219
src/AmmServerlib/stringscanners/firstLines.c, 220
src/AmmServerlib/stringscanners/firstLines.h, 221
src/AmmServerlib/stringscanners/httpHeader.c, 222
src/AmmServerlib/stringscanners/httpHeader.h, 223
src/AmmServerlib/stringscanners/imageFiles.c, 224
src/AmmServerlib/stringscanners/imageFiles.h, 224
src/AmmServerlib/stringscanners/postHeader.c, 227
src/AmmServerlib/stringscanners/postHeader.h, 227
src/AmmServerlib/stringscanners/textFiles.c, 228
src/AmmServerlib/stringscanners/textFiles.h, 229
src/AmmServerlib/stringscanners/videoFiles.c, 230
src/AmmServerlib/stringscanners/videoFiles.h, 231
src/AmmServerlib/threads/clientServer.c, 233
src/AmmServerlib/threads/clientServer.h, 234
src/AmmServerlib/threads/freshThreads.c, 235
src/AmmServerlib/threads/freshThreads.h, 236
src/AmmServerlib/threads/prespawnedThreads.c, 237
src/AmmServerlib/threads/prespawnedThreads.h, 239
src/AmmServerlib/threads/threadInitHelper.c, 245
src/AmmServerlib/threads/threadInitHelper.h, 246
src/AmmServerlib/threads/threadedServer.c, 240
src/AmmServerlib/threads/threadedServer.h, 243
src/AmmServerlib/tools/directory_lists.c, 246
src/AmmServerlib/tools/directory_lists.h, 247
src/AmmServerlib/tools/http_tools.c, 249
src/AmmServerlib/tools/http_tools.h, 256
src/AmmServerlib/tools/logs.c, 262
src/AmmServerlib/tools/logs.h, 263
src/AmmServerlib/tools/time_provider.c, 265
src/AmmServerlib/tools/time_provider.h, 267
src/AmmServerlib/version.h, 268
src/ScriptRunner/main.c, 78
src/Services/AmmarServer/main.c, 82
src/Services/CinemaPilot/main.c, 85
src/Services/GeoPosShare/main.c, 88
src/Services/HabChan/board.c, 269
src/Services/HabChan/board.h, 269
src/Services/HabChan/main.c, 91
src/Services/HabChan/main.h, 270
src/Services/HabChan/postReceiver.c, 270
src/Services/HabChan/postReceiver.h, 270
src/Services/HabChan/state.c, 271
src/Services/HabChan/state.h, 272
src/Services/HabChan/thread.c, 274
src/Services/HabChan/thread.h, 275
src/Services/MyBlog/database.c, 276
src/Services/MyBlog/database.h, 277
src/Services/MyBlog/index.c, 278
src/Services/MyBlog/index.h, 279
src/Services/MyBlog/main.c, 92
src/Services/MyBlog/tools/myblogTool.c, 280
src/Services/MyLoader/main.c, 94
src/Services/MyRemoteDesktop/main.c, 95
src/Services/MyRemoteDesktop/xwd-1.0.5/XwdLib.h, 298
src/Services/MyRemoteDesktop/xwd-1.0.5/clientwin.c, 281
src/Services/MyRemoteDesktop/xwd-1.0.5/clientwin.h, 281
src/Services/MyRemoteDesktop/xwd-1.0.5/config.h, 281
src/Services/MyRemoteDesktop/xwd-1.0.5/dsimple.c, 283
src/Services/MyRemoteDesktop/xwd-1.0.5/dsimple.h, 284
src/Services/MyRemoteDesktop/xwd-1.0.5/list.c, 286
src/Services/MyRemoteDesktop/xwd-1.0.5/list.h, 288

src/Services/MyRemoteDesktop/xwd-1.0.5/main.c, 97
src/Services/MyRemoteDesktop/xwd-1.0.5/multiVis.c, 292
src/Services/MyRemoteDesktop/xwd-1.0.5/multiVis.h, 294
src/Services/MyRemoteDesktop/xwd-1.0.5/wsutils.h, 295
src/Services/MyRemoteDesktop/xwd-1.0.5/xwd.c, 297
src/Services/MyTube/indexer.c, 299
src/Services/MyTube/indexer.h, 299
src/Services/MyTube/main.c, 99
src/Services/MyTube/thumbnailer.c, 300
src/Services/MyTube/thumbnailer.h, 301
src/Services/MyURL/main.c, 102
src/Services/SQLiteServer/main.c, 108
src/Services/SQLiteServer/sqlite.c, 305
src/Services/SQLiteServer/sqlite.h, 305
src/Services/ScriptRunner/main.cpp, 301
src/Services/SimpleTemplate/main.c, 106
src/StringRecognizer/fastStringParser.c, 306
src/StringRecognizer/fastStringParser.h, 308
src/StringRecognizer/main.c, 109
src/UnitTests/testHashMap.c, 310
src/UserAccounts/main.c, 110
src/UserAccounts/userAccounts.h, 311
start_timer
    time_provider.c, 266
    time_provider.h, 268
StartHTTPServer
    threadedServer.c, 241
    threadedServer.h, 245
startMplayer
    Services/CinemaPilot/main.c, 88
starting
    directory_lists.c, 247
state.h
    FILETYPE_AUDIO, 273
    FILETYPE_FORBIDDEN, 273
    FILETYPE_IMAGE, 273
    FILETYPE_VIDEO_FILE, 273
    FILETYPE_VIDEO_YOUTUBE, 273
    NUMBER_OF_FILETYPES, 273
state.c
    addPostToThread, 271
    admin_server, 272
    boardHashMap, 272
    debug_get_callback, 271
    default_server, 272
    GET_override, 272
    loadSite, 271
    ourSite, 272
    threadHashMap, 272
    threadIndexEndPage, 272
    threadIndexEndPageLength, 272
    threadIndexPage, 272
    threadIndexPageLength, 272
    threadIndexStartPage, 272
    threadIndexStartPageLength, 272
    unloadSite, 271
state.h
    addPostToThread, 274
    admin_server, 274
    boardHashMap, 274
    default_server, 274
    FILETYPES_ENUM, 273
    GET_override, 274
    LINE_MAX_LENGTH, 273
    loadSite, 274
    MAX_BOARDS, 273
    MAX_STRING_SIZE, 273
    ourSite, 274
    threadHashMap, 274
    threadIndexEndPage, 274
    threadIndexEndPageLength, 274
    threadIndexPage, 274
    threadIndexPageLength, 274
    threadIndexStartPage, 274
    threadIndexStartPageLength, 274
    unloadSite, 274
stateType
    Services/CinemaPilot/main.c, 86
stats
    main.cpp, 304
    ScriptRunner/main.c, 81
    Services/AmmarServer/main.c, 84
    Services/CinemaPilot/main.c, 88
    Services/MyBlog/main.c, 94
    Services/MyLoader/main.c, 95
    Services/SimpleTemplate/main.c, 107
    Services/SQLiteServer/main.c, 109
sticky
    thread, 47
stop
    Services/AmmarServer/main.c, 84
stop_callback
    Services/AmmarServer/main.c, 84
stop_server
    AmmServer_Instance, 18
StopHTTPServer
    threadedServer.c, 243
    threadedServer.h, 245
stopMplayer
    Services/CinemaPilot/main.c, 88
stopTime
    playlistItem, 41
store_new_configuration_callback
    main.cpp, 304
    ScriptRunner/main.c, 81
str
    fspString, 24
    InputParserC, 35
Str2Int_internal
    InputParser_C.c, 189
str_length
    InputParserC, 35
strIDFriendly

fspString, 24
strLength
fspString, 24
strToUpcase
http_tools.c, 256
http_tools.h, 262
StringIsHTMLSafe
main.cpp, 304
StringRecognizer/main.c
main, 109
stringsLoaded
fastStringParser, 24
StripGETRequestQueryAndFragment
http_tools.c, 255
http_tools.h, 261
StripHTMLCharacters_Inplace
http_tools.c, 255
http_tools.h, 261
StripVariableFromGETorPOSTString
http_tools.c, 255
http_tools.h, 262
stristr
http_tools.c, 256
stristr2Caps
http_tools.c, 256
strlimcpy
index.c, 279
struct_cmp_urldb_items
Services/MyURL/main.c, 105
supports_compression
HTTPHeader, 29

TEXT
http_tools.h, 257
TEXTFILES_CSS
textFiles.h, 229
TEXTFILES_DOC
textFiles.h, 230
TEXTFILES_EMPTY
textFiles.h, 229
TEXTFILES_END_OF_ITEMS
textFiles.h, 230
TEXTFILES_HTM
textFiles.h, 229
TEXTFILES_HTML
textFiles.h, 229
TEXTFILES_ODF
textFiles.h, 230
TEXTFILES_ODT
textFiles.h, 230
TEXTFILES_RTF
textFiles.h, 230
TEXTFILES_TXT
textFiles.h, 229
TRACE
AmmServerlib.h, 121
TRUE_COLOR
multiVis.c, 293
tag
tagItem, 46
tag_after_image
directory_lists.c, 247
tag_pre_image
directory_lists.c, 247
tagHash
tagItem, 46
tagItem, 46
tag, 46
tagHash, 46
tagItemList, 46
currentTags, 47
item, 47
maxTags, 47
tags
postItem, 43
tagsStr
videoItem, 52
target_win
Services/MyRemoteDesktop/xwd-1.0.5/main.c, 99
templates_root
AmmServer_Instance, 18
helloworld.c, 56
main.cpp, 304
PreSpawnedThread, 44
ScriptRunner/main.c, 81
Services/AmmarServer/main.c, 85
Services/CinemaPilot/main.c, 88
Services/GeoPosShare/main.c, 91
Services/HabChan/main.c, 92
Services/MyBlog/main.c, 94
Services/MyLoader/main.c, 95
Services/MyRemoteDesktop/main.c, 97
Services/MyTube/main.c, 101
Services/MyURL/main.c, 106
Services/SimpleTemplate/main.c, 107
Services/SQLiteServer/main.c, 109
TemplatesInternalURI
server_configuration.c, 204
server_configuration.h, 214
term_buffer
jpgInput.c, 114
termination_handler
main.cpp, 304
ScriptRunner/main.c, 81
TerminationCallback
AmmServerlib/main.c, 78
testAmmCaptcha
AmmCaptcha.h, 57
AmmCaptcha/main.c, 59
testHashMap.c
BLACK, 310
BLUE, 310
CYAN, 310
doHashMapTest, 311
doInjectTest, 311
GREEN, 310
MAGENTA, 311

- main, 311
- NORMAL, 311
- RED, 311
- WHITE, 311
- YELLOW, 311

textFiles.h
- TEXTFILES_CSS, 229
- TEXTFILES_DOC, 230
- TEXTFILES_EMPTY, 229
- TEXTFILES_END_OF_ITEMS, 230
- TEXTFILES_HTM, 229
- TEXTFILES_HTML, 229
- TEXTFILES_ODF, 230
- TEXTFILES_ODT, 230
- TEXTFILES_RTF, 230
- TEXTFILES_TXT, 229

textFiles.c
- scanFor_textFiles, 229

textFiles.h
- scanFor_textFiles, 230

thread, 47
- creation, 47
- lastReply, 47
- maxNumberOfReplies, 47
- numberOfImages, 47
- numberOfReplies, 47
- op, 47
- password, 47
- repliable, 47
- replies, 47
- sticky, 47
- title, 47

thread.c
- addThreadToBoard, 275
- loadThread, 275
- mallocHTMLListOfThreadsOfBoard, 275
- prepareThreadIndexView, 275
- prepareThreadView, 275

thread.h
- addThreadToBoard, 275
- prepareThreadIndexView, 275
- prepareThreadView, 275

thread_id
- PassToHTTPThread, 40
- PreSpawnedThread, 44

threadHashMap
- state.c, 272
- state.h, 274

threadID
- HTTPTransaction, 30

threadIndexEndPage
- state.c, 272
- state.h, 274

threadIndexEndPageLength
- state.c, 272
- state.h, 274

threadIndexPage
- state.c, 272
- state.h, 274

threadIndexPageLength
- state.c, 272
- state.h, 274

threadIndexStartPage
- state.c, 272
- state.h, 274

threadIndexStartPageLength
- state.c, 272
- state.h, 274

threadIndexView
- Services/HabChan/main.c, 92

threadNum
- PreSpawnedThread, 44

threadQueue
- board, 22

threadUID
- board, 22

threadView
- Services/HabChan/main.c, 92

threadedServer.c
- HTTPServerIsRunning, 241
- MainHTTPServerThread, 241
- StartHTTPServer, 241
- StopHTTPServer, 243

threadedServer.h
- HTTPServerIsRunning, 244
- StartHTTPServer, 245
- StopHTTPServer, 245

threads
- board, 22

threads_pool
- AmmServer_Instance, 18

thumbnail
- videoItem, 52

thumbnailAllVideoDatabase
- Services/MyTube/main.c, 101

thumbnailContext
- Services/MyTube/main.c, 102

thumbnailer.c
- generateThumbnailOfVideo, 301

thumbnailer.h
- generateThumbnailOfVideo, 301

time_provider.c
- days, 266
- end_timer, 265
- GetDateString, 266
- GetTickCountAmmServ, 266
- months, 266
- start_timer, 266

time_provider.h
- end_timer, 267
- GetDateString, 267
- GetTickCountAmmServ, 268
- start_timer, 268

time_snap, 48
- difference, 48

timestamp, 48

day, 49
hour, 49
minute, 49
month, 49
second, 49
wday, 49
year, 49
title
postItem, 43
thread, 47
videoItem, 52
token_start
tokens, 49
tokenlist
InputParserC, 35
tokens, 49
length, 49
token_start, 49
tokens_count
InputParserC, 35
tokens_max
InputParserC, 35
top_image
main.cpp, 304
ScriptRunner/main.c, 81
totalDataLength
htmlContent, 27
TransmitFileToSocket
file_server.c, 195
TransmitFileToSocketInternal
file_server.c, 195
transparentType
OverlayInfo, 38
OverlayVisualPropertyRec, 39
triggerTime
playlistItem, 41
trim_last_empty_chars
http_tools.c, 256
http_tools.h, 262
twitterURL
socialLinks, 45
TypesOfRequests
AmmServerlib.h, 121

URLDB, 50
longURL, 50
shortURL, 50
shortURLHash, 50
USE_BINARY_SEARCH
Services/MyURL/main.c, 104
USE_JPG_FILES
jpgInput.h, 115
USE_SCANF
InputParser_C.h, 191
USERNAME
AmmServer_Instance_Settings, 19
uadb_authenticateUser
userAccounts.h, 312
UserAccounts/main.c, 110
uadb_closeUserAccountDatabase
userAccounts.h, 312
UserAccounts/main.c, 110
uadb_initializeUserAccountDatabase
userAccounts.h, 312
UserAccounts/main.c, 110
uadb_loginUser
userAccounts.h, 312
UserAccounts/main.c, 110
unloadSite
state.c, 271
state.h, 274
uploadProcessor
Services/MyLoader/main.c, 95
usage
dsimple.h, 286
Services/MyRemoteDesktop/xwd-1.0.5/main.c, 98
xwd.c, 298
UsePreSpawnedThreadToServeNewClient
prespawnedThreads.c, 239
prespawnedThreads.h, 240
userAccounts.h
ENCODING_AVAILIABLE_TYPES, 312
ENCODING_PLAINTEXT, 312
ENCODING_SHA1, 312
UserAccount_PasswordEncoding
userAccounts.h, 312
UserAccount_UserID
userAccounts.h, 312
UserAccountAuthenticationToken, 50
dummy, 50
UserAccountDatabase, 50
dummy, 51
lastAuthenticationToken, 51
UserAccountPasswordEncodingEnum
userAccounts.h, 312
userAccounts.h
uadb_authenticateUser, 312
uadb_closeUserAccountDatabase, 312
uadb_initializeUserAccountDatabase, 312
uadb_loginUser, 312
UserAccount_PasswordEncoding, 312
UserAccount_UserID, 312
UserAccountPasswordEncodingEnum, 312
UserAccounts/main.c
uadb_authenticateUser, 110
uadb_closeUserAccountDatabase, 110
uadb_initializeUserAccountDatabase, 110
uadb_loginUser, 110
userAgent
HTTPHeader, 29
userAgentLength
HTTPHeader, 29
userList
clientListContext, 23

VIDEO
http_tools.h, 258
VIDEOFILES_3GP

videoFiles.h, 231
VIDEOFILES_AVI
videoFiles.h, 231
VIDEOFILES_EMPTY
videoFiles.h, 231
VIDEOFILES_END_OF_ITEMS
videoFiles.h, 231
VIDEOFILES_FLV
videoFiles.h, 231
VIDEOFILES_H263
videoFiles.h, 231
VIDEOFILES_H264
videoFiles.h, 231
VIDEOFILES_MKV
videoFiles.h, 231
VIDEOFILES_MP4
videoFiles.h, 231
VIDEOFILES_MPEG
videoFiles.h, 231
VIDEOFILES_MPEG4
videoFiles.h, 231
VIDEOFILES_WEBM
videoFiles.h, 231
VERSION
config.h, 282
VIDEO_FILES_PATH_1
Services/MyTube/main.c, 100
VIDEO_FILES_PATH_2
Services/MyTube/main.c, 100
VIDEO_FILES_PATH_3
Services/MyTube/main.c, 100
value
OverlayInfo, 38
OverlayVisualPropertyRec, 39
varSocketTimeoutREAD_seconds
server_configuration.c, 204
server_configuration.h, 214
varSocketTimeoutWRITE_seconds
server_configuration.c, 204
server_configuration.h, 214
ver
InputParser.cpp, 186
verified_local_resource
HTTPHeader, 29
Version
InputParser, 33
InputParser.cpp, 186
version.h
RC_FILEVERSION, 268
video
videoCollection, 51
videoFiles.h
VIDEOFILES_3GP, 231
VIDEOFILES_AVI, 231
VIDEOFILES_EMPTY, 231
VIDEOFILES_END_OF_ITEMS, 231
VIDEOFILES_FLV, 231
VIDEOFILES_H263, 231
VIDEOFILES_H264, 231
VIDEOFILES_MKV, 231
VIDEOFILES_MP4, 231
VIDEOFILES_MPEG, 231
VIDEOFILES_MPEG4, 231
VIDEOFILES_WEBM, 231
video_root
Services/MyTube/main.c, 102
videoCollection, 51
MAX_numberOfVideos, 51
numberOfLoadedVideos, 51
video, 51
videoDefaultTestTranmission
indexer.c, 299
indexer.h, 300
videoFileContext
Services/MyTube/main.c, 102
videoFiles.c
scanFor_videoFiles, 230
videoFiles.h
scanFor_videoFiles, 232
videoItem, 51
comment, 52
dislikes, 52
filename, 52
hashID, 52
likes, 52
tagsStr, 52
thumbnail, 52
title, 52
views, 52
visibility, 52
videoPageContext
Services/MyTube/main.c, 102
views
videoItem, 52
vis
image_region_type, 31
image_win_type, 31
visibility
videoItem, 52
visible_region
image_region_type, 31
visualID
OverlayVisualPropertyRec, 39

WEBSERVERROOT
Services/AmmarServer/main.c, 83
Services/HabChan/main.c, 92
WHITE
AmmServerlib/InputParser/InputParser_C_-Tester/main.c, 60
logs.h, 264
testHashMap.c, 311
warning
logs.c, 263
logs.h, 265
warningsAboutIncorrectlyAllocatedStackIssued
InputParser_C.c, 189

warpImage
    img_warp.c, 112
    img_warp.h, 113
wday
    timestamp, 49
web_root_path
    AmmServer_RH_Context, 21
webserver_root
    AmmServer_Instance, 18
    helloworld.c, 56
    main.cpp, 304
    PreSpawnedThread, 44
    ScriptRunner/main.c, 82
    Services/AmmarServer/main.c, 85
    Services/CinemaPilot/main.c, 88
    Services/GeoPosShare/main.c, 91
    Services/HabChan/main.c, 92
    Services/MyBlog/main.c, 94
    Services/MyLoader/main.c, 95
    Services/MyRemoteDesktop/main.c, 97
    Services/MyTube/main.c, 102
    Services/MyURL/main.c, 106
    Services/SimpleTemplate/main.c, 108
    Services/SQLiteServer/main.c, 109
website, 52
    allowComments, 52
    allowPing, 52
    blogTitle, 52
    linksLeft, 53
    linksRight, 53
    menu, 53
    post, 53
    siteDescription, 53
    siteName, 53
    siteURL, 53
    social, 53
    widget, 53
widget
    website, 53
widgetItem, 53
    content, 53
    label, 53
    link, 53
widgetItemList, 53
    currentItems, 54
    item, 54
    maxItems, 54
width
    Image, 30
    image_region_type, 31
    image_win_type, 32
win
    image_region_type, 31
    image_win_type, 32
Window_Dump
    Services/MyRemoteDesktop/xwd-1.0.5/main.c, 99
    xwd.c, 298
Window_With_Name
    dsimple.c, 284
    dsimple.h, 286
WriteJPEGFile
    jpgInput.c, 114
    jpgInput.h, 115
WriteJPEGInternal
    jpgInput.c, 114
WriteJPEGMemory
    jpgInput.c, 115
    jpgInput.h, 115
WritePPM
    imaging.c, 111
    imaging.h, 112
wsutils.h
    CreateImagePlanesWindow, 296
    CreateOverlayPlanesWindow, 296
    FLEXIBLE, 296
    FindImagePlanesVisual, 297
    FindOverlayPlanesVisual, 297
    FreeXVisualInfo, 297
    GetXVisualInfo, 297
    NOT_FLEXIBLE, 296
    None, 296
    SB_CMAP_TYPE_FULL, 296

x1
    myBox, 37
x2
    myBox, 37
X_USAGE
    dsimple.h, 285
x_rootrel
    image_region_type, 31
    image_win_type, 32
x_vis
    image_region_type, 31
    image_win_type, 32
XWDLIB_BRIDGE
    Services/MyRemoteDesktop/main.c, 96
xwd.c
    _swaplong, 298
    _swapshort, 298
    FEEP_VOLUME, 298
    Get_XColors, 298
    Image_Size, 298
    lowbit, 298
    main, 298
    Pixel, 298
    usage, 298
    Window_Dump, 298
XwdLib.h
    closeXwdLib, 298
    getScreen, 298
    initXwdLib, 298

y1
    myBox, 37
y2
    myBox, 37

y_rootrel
    image_region_type, 31
    image_win_type, 32
y_vis
    image_region_type, 31
    image_win_type, 32
YELLOW
    AmmServerlib/InputParser/InputParser_C_Tester/main.c, 60
    AString.c, 135
    logs.h, 264
    testHashMap.c, 311
year
    timestamp, 49
youtubeURL
    socialLinks, 45

zero_list
    list.c, 288
    list.h, 291