SRP_HTTP_FRAMEWORK_SETUP

Stores the setup information used by the SRP HTTP Framework. The layout looks like this:

Attribute	Name	Description
<1>	**Home URL**	Identifies the primary URL of the website, e.g., *www.mywebsite.com*.
<2>	**API URL**	Identifies the path that is appended to the Home URL wherein the entry point of the API begins, e.g., */api* The HTTP_Service_Setup uses the above information to create proper RESTful responses wherein paths to other web service APIs need to be included within the response. Thus, in the above examples, the two are joined when creating fully resolved API URLs, e.g., *www.mywebsite.com/api/<service>*
<3>	**Capture Path**	Used in HTTP_MCP to identify the local OS path where request and response content can be saved for off-line analysis. If the path does not exist, then no attempt to store this information will be made.
<4>	**Enable Authentication Flag**	Boolean setting that determines if authentication is enabled. Used by HTTP_Authentication_Services. Note: Only an explicit value of 0 (*False*) will disable authentication. Any other value, including an empty value, will enable authentication.
<5>	**Realm Value**	Used with HTTP authentication to associate resource links to the same protective space.
<6>	**Entry Point Service**	Name of the entry point web service. This is used by HTTP_MCP to call the first web service. This should not include the standard HTTP prefix or SERVICES suffix. The default value is *entry_point*.
<7>	**Flush Cache Flag**	Boolean setting that determines if code should be flushed when the web service is finished. This allows changes in web service procedures to always be available without having to restart the OEngineServer.
<8>	**Non-Authenticated URLs**	@VM list of URLs that should not be authenticated. This includes URLs that support OAuth redirects since these requests will unlikely be authenticated (although they should be secret and secure with the OAuth authenticating server).
<9>	**Aborted Service**	Service handler for HTTP Requests that get aborted either due to Runtime Errors or Status Errors. If the Debugger Intercept has been enabled, this handler will only receive Status Error aborts.
<10>	**Enable Logging Flag**	Boolean setting that determines if logging is enabled. This is used by the *CreateLogFile* service. Note: Only an explicit value of 0 (*False*) will disable logging. Any other value, including an empty value, will enable logging.
<11>	**Debugger Setting**	Debugger Setting value that will be passed into the RTI_Debugger_Setting subroutine. 0=Disabled, 1=Enabled, and 2=Intercept. If value is 2, the value of HTTP_FRAMEWORK_SETUP_DEBUGGER_INTERCEPT$ will be used to determine the name of the intercept stored procedure.
<12>	**Debugger Intercept**	Service handler for HTTP Requests that abort due to a Runtime Error and the Debugger Intercept has been enabled. See HTTP_FRAMEWORK_SETUP_DEBUGGER_SETTING$.
<13>	**Log Errors Only**	Boolean setting that determines if only error responses (i.e., status codes of 4xx or 5xx) should be logged. This is used by the *CreateLogFile* service. Note: Only an explicit value of 0 (False) will disable logging. Any other value, including an empty value, will enable logging.
<14>	**Setup Whitelisted IPs**	@VM list of IPs that should be permitted. If this list is empty, then all IPs are permitted. Note, a valid IP does not automatically mean the request is authenticated. That is a separate check.
<15>	**Enable HTTP Basic Authentication**	Boolean setting that determines if HTTP Basic Authentication is enabled. Used by HTTP_Authentication_Services. This is ignored if the Enable Authentication flag is set to False.
<16>	**New Password Time to Live**	If HTTP Basic Authentication is enabled, this is how long (in hours) new passwords that are created can be valid before needing to be reset. If left empty, passwords do not expire.
<17>	**Old Password Time to Live**	If HTTP Basic Authentication is enabled, this is how long (in hours) old passwords can remain valid. This should be a short period of time allowing for the new password to propogate. If left empty, old passwords will only be valid for 1 hour.
<18>	**Invalid Password Limit**	If HTTP Basic Authentication is enabled, this is how many attempts to access the system with an invalid password will be allowed before containment action is taken.
<19>	**Containment Action**	If HTTP Basic Authentication is enabled, this determines the containment action to take when the number of invalid password attempts has been exceeded.
<20>	**API Call Procedure**	The type of method used to call the API.

<21>	**Non-Authenticated Query Params**	@VM/@TM delimited list of query params that should be used to further define which URL paths should be non-authenticated.
<22>	**Banned IPs**	@VM list of IPs that should be banned.
<23>	**Whitelisted IPs Type**	Flag to determine if Whitelisted IPs should represent only those IPs that will be permitted (default) or if Whitelisted IPs should always be permitted (i.e., do not require other forms of authentication). Empty value or 1 means restricted IP access, 2 means IPs are always permitted.

Note: `SRP_HTTP_FRAMEWORK_SETUP` is the default configuration row for all applications. Each local application will copy this and maintain a local version using the format `SRP_HTTP_FRAMEWORK_SETUP*<AppID>`.