

HTTP_JSON_Services

Application service module that helps build responses using various JSON formats.

Syntax

```
Response = HTTP_JSON_Services(@Service, @Params)
```

Returns

The meaning of the response value depends on the service.

Parameters

Parameter	Description
@Service	The name of the service being requested. Required.
@Params	Generic parameters. Refer to a specific service to determine the actual parameters used.

Remarks

The SRP HTTP Framework supports two popular JSON friendly data formats: [HAL](#) and [Schema](#). HAL stands for *Hypermedia Application Language* and is an emerging format that implements the [HATEOAS](#) constraint of REST. There are other formats that are also emerging and none hold dominance at this point. Likewise, JSON itself stands alongside XML as a widely accepted general purpose format for representing resources. Therefore, if you need or prefer to use another format then you are welcome to extend [HTTP_JSON_Services](#) (in case you want to use a different hypermedia format, such as [Collection](#)) or write a new service such as [HTTP_XML_Services](#) (in case you want to use a different data format, such as XML).

Schema is a meta-data format. It is used to notify clients what kind of data is available, which is handy when user interfaces need to be dynamically built. Schema can be used to define if a prompt is text, numeric, date, combobox, Boolean, etc. Like HAL, Schema is also an emerging format. There are others that may be better suited to your needs.

Services

Service	Description
SethHALItem	Usage: HTTP_JSON_Services('SethHALItem', ItemURL, ColumnNames, ColumnValues, DataTypes) Comments: Creates a HAL+JSON object for a specific item. Requires the GetHAL service to return the serialized object. Returns: N/A
SethHALCollection	Usage: HTTP_JSON_Services('SethHALCollection', CollectionURL, ItemsURLs, ItemsTitles) Comments: Creates a HAL+JSON object for a collection. Requires the GetHAL service to return the serialized object. Returns: N/A
SethHALCollectionEmbedded	Usage: HTTP_JSON_Services('SethHALCollectionEmbedded', CollectionURL, ItemsURLs, ColumnNames, ColumnValues, DataTypes) Comments: Creates a HAL+JSON object for a collection of embedded items. Requires the GetHAL service to return the serialized object. Returns: N/A

SetHALLinks	Usage: HTTP_JSON_Services('SetHALLinks', SelfURI, HREFNames, HREFURIs, ChildNames, hChildren, Names, Values) Comments: Creates a HAL style "links" object. Returns: N/A
GetHAL	Usage: HTTP_JSON_Services('GetHAL', ItemArrayLabel) Comments: Returns the serialized JSON object for the current HAL response. If no HAL object has been defined then this will return an empty string and a 500 status code will be set. All HAL objects and arrays will be released in this service. Returns: A serialized HAL+JSON object based based on the current HAL objects in memory.
GetSchemaRootObj	Usage: HTTP_JSON_Services('GetSchemaRootObj') Comments: Returns the handle to a root Schema object. If it does not already exist it will be created with the standard "\$schema" value already added. Returns: Handle to a root Schema+JSON object.
SetSchemaMeta	Usage: HTTP_JSON_Services('SetSchemaMeta', Title, Description, Type) Comments: Sets the meta data associated to the schema object. Returns: N/A
SetSchemaProperty	Usage: HTTP_JSON_Services('SetSchemaProperty', Name, Title, Type, Format, EnumList, Required) Comments: Sets a property to the schema. There can be more than one property so this service will add another property to the list if it already exists. Returns: N/A
GetSchema	Usage: HTTP_JSON_Services('GetSchema') Comments: Returns the serialized JSON object for the current schema. If no schema object has been defined then this will return an empty string and a 500 status code will be set. All schema objects and arrays will be released in this service. Returns: A serialized Schema+JSON object based based on the current Schema objects in memory.
GetURLFromID	Usage: HTTP_JSON_Services('GetURLFromID', ID) Comments: Returns a URL segment for the ID passed into the service. This creates a "slug" style URL so that it will be search friendly, human readable, and an RESTful. Returns: A "slug" style URL segment for a given resource ID.
GetIDFromURL	Usage: HTTP_JSON_Services('GetIDFromURL', URL, Array) Comments: Returns the ID based on the URL passed into the service. This service attempts to reverse engineer the URL. Returns: A resource (item) ID.

Params

The proper use of the generic arguments are defined in the definition of each service above.