ifUrlTransfer

Implemented By

- roUrlTransfer

Supported Methods

- GetIdentity() as Integer
- SetUrl(url as String) as Void
- GetUrl() as String
- SetRequest(req as String)
- GetRequest() as String
- GetToString() as String
- GetToFile(filename as String) as Integer
- AsyncGetToString() as Boolean
- AsyncGetToFile(filename as String) as Boolean
- Head() as Dynamic
- AsyncHead() as Boolean
- PostFromString(request as String) as Integer
- PostFromFile(filename as String) as Integer
- AsyncPostFromString(request as String) as Boolean
- AsyncPostFromFile(filename as String) as Boolean
- AsyncPostFromFileToFile(fromFile as String, toFile as String) as Boolean
- AsyncCancel() as Boolean
- RetainBodyOnError(retain as Boolean) as Boolean
- SetUserAndPassword(user as String, password as String) as Boolean
- SetMinimumTransferRate(bytes_per_second as Integer, period_in_seconds as Integer) as Boolean
- GetFailureReason() as String
- EnableEncodings(enable as Boolean) as Boolean
- Escape(text as String) as String
- Unescape(text as String) as String
- UrlEncode(url as String) as String
- EnableResume(enable as Boolean) as Boolean
- EnablePeerVerification(enable as Boolean) as Boolean
- EnableHostVerification(enable as Boolean) as Boolean
- EnableFreshConnection(enable as Boolean) as Boolean
- SetHttpVersion(version as String) as Void

Note on asynchronous methods

Each roUrlTransfer object can perform only one asynchronous operation at one time. After starting an asynchronous operation, you cannot perform any other data transfer operations using that object until the asynchronous operation has completed, as indicated by receiving an roUrlEvent message whose GetSourceIdentity value matches the GetIdentity value of the roUrlTransfer. Furthermore, the roUrlTransfer object must remain referenced until the transfer has completed. That means that there must be at least one variable containing a reference to the object during the transfer. Allowing the variable to go out of scope (for example, by returning from a function where the variable is declared, or reusing the variable to hold a different value) will stop the asynchronous transfer.

Description of Methods

GetIdentity() as Integer

Returns a unique number for this object that can be used to identify whether events originated from this object.

Note that the value can be any arbitrary value as assigned by the firmware, and should only be used for comparison purposes.
For example, the value should not be used as an array index. For use as a look-up key, one option would be to use GetIdentity().ToStr() as an associative array key.

Example code

```
Function Setup()
    m.pendingXfers = {}
End Function

Function GetAsync(url as String)
    newXfer = CreateObject("roUrlTransfer")
    newXfer.SetUrl(url)
    newXfer.AsyncGetToString()
    requestId = newXfer.GetIdentity().ToStr()
    m.pendingXfers[requestId] = newXfer
End Function

Function HandleUrlEvent(event as Object)
    requestId = event.GetSourceIdentity().ToStr()
    xfer = m.pendingXfers[requestId]
    if xfer <> invalid then
        ' process it
        m.pendingXfers.Delete(requestId)
    end if
End Function
```

SetUrl(url as String) as Void

Sets the URL to use for the transfer request.

GetUrl() as String

Returns the current URL.

SetRequest(req as String)

Changes the request method from the normal GET, HEAD or POST to the value passed as a string. This should be used with caution as it can generate invalid HTTP requests.

GetRequest() as String

Returns the current request method.

GetToString() as String

Connect to the remote service as specified in the URL and return the response body as a string. This function waits for the transfer to complete and it may block for a long time.

This calls discards the headers and response codes. If that information is required, use AsyncGetToString instead.

GetToFile(filename as String) as Integer

Connect to the remote service as specified in the URL and write the response body to a file on the Roku device's filesystem.

This function does not return until the exchange is complete and may block for a long time.

The HTTP response code from the server is returned. It is not possible to access any of the response headers. If this information is required use AsyncGetToFile instead.

AsyncGetToString() as Boolean

Start a GET request to a server, but do not wait for the transfer to complete. When the GET completes, a roUrlEvent will be sent to the message port associated with the object. The event will contain a roString with the body of the response. If false is returned then the request could not be issued and no events will be delivered.

AsyncGetToFile(filename as String) as Boolean

Like AsyncGetToString, this starts a transfer without waiting for it to complete. However, the response body will be written to a file on the device's filesystem instead of being returned in a String object. When the GET completes, an roUrlEvent will be sent to the message port associated with the object. If false is returned then the request could not be issued and no events will be delivered.

Head() as Dynamic

Synchronously perform an HTTP HEAD request and return an roUrlEvent object. In the event of catastrophic failure (e.g. an asynchronous operation is already active) then **invalid** is returned.

AsyncHead() as Boolean

Begin an HTTP HEAD request without waiting for it to complete. When the HEAD completes, an roUrlEvent will be sent to the message port associated with the object. If false is returned then the request could not be issued and no events will be delivered.

PostFromString(request as String) as Integer

Use the HTTP POST method to send the supplied string to the current URL. The HTTP response code is returned. Any response body is discarded.

PostFromFile(filename as String) as Integer

Use the HTTP POST method to send the contents of the specified file to the current URL. The HTTP response code is returned. Any response body is discarded.

AsyncPostFromString(request as String) as Boolean

Use the HTTP POST method to send the supplied string to the current URL. When the POST completes, an roUrlEvent will be sent to the message port associated with the object. If false is returned then the request could not be issued and no events will be delivered.

AsyncPostFromFile(filename as String) as Boolean

Use the HTTP POST method to send the contents of the specified file to the current URL. When the POST completes, an roUrlEvent will be sent to the message port associated with the object. If false is returned then the request could not be issued and no events will be delivered.

AsyncPostFromFileToFile(fromFile as String, toFile as String) as Boolean

Use the HTTP POST method to send the contents of the specified file (fromFile) to the current URL. When the POST completes successfully, an roUrlEvent will be sent to the message port associated with the object. If false is returned then the request could not be issued and no events will be delivered. This function is the same as AsyncPostFromFile, except that the HTTP response is written to the file specified by the toFile parameter.

AsyncCancel() as Boolean

Cancel any outstanding async requests on the roUrlTransfer object.

RetainBodyOnError(retain as Boolean) as Boolean

If retain is true, return the body of the response even if the HTTP status code indicates that an error occurred.

SetUserAndPassword(user as String, password as String) as Boolean

Enables HTTP authentication using the specified user name and password. Note that HTTP basic authentication is deliberately disabled due to it being inherently insecure. HTTP digest authentication is supported.

SetMinimumTransferRate(bytes_per_second as Integer, period_in_seconds as Integer) as Boolean

Terminate the transfer automatically if the rate drops below bytes_per_second when averaged over period_in_seconds.If the transfer is over the Internet you may not want to set period_in_seconds to a small number because network problems may cause temporary drops in performance. For large file transfers and a small bytes_per_second, averaging over fifteen minutes or even longer might be appropriate.

GetFailureReason() as String

If any of the roUrlTransfer functions indicate failure then this function may provide more information regarding the failure.

EnableEncodings(enable as Boolean) as Boolean

Enable gzip encoding of transfers.

Escape(text as String) as String

URL encode the specified string per RFC 3986 and return the encoded string.

Unescape(text as String) as String

Decode the specified string per RFC 3986 and return the unencoded string.

UrlEncode(url as String) as String

Same as Escape. *This method is deprecated.*

EnableResume(enable as Boolean) as Boolean

Enable automatic resumption of AsyncGetToFile and GetToFile requests.

EnablePeerVerification(enable as Boolean) as Boolean

Verify the certificate has a chain of trust up to a valid root certificate using. CURLOPT_SSL_VERIFYPEER.

EnableHostVerification(enable as Boolean) as Boolean

Verify that the certificate belongs to the host we're talking to using CURLOPT_SSL_VERIFYHOST.

EnableFreshConnection(enable as Boolean) as Boolean

Enable fresh connection using CURLOPT_FRESH_CONNECT.

SetHttpVersion(version as String) as Void

Available since firmware version 7.6

This is an optional function to enable HTTP/2 support. If `version` is set to `"http2"`, HTTP/2 will be used for all underlying transfers. This must be set on a `roUrlTransfer` instance prior to any data transfer. This cannot be changed on an instance after it is used.

Note: For the HTTP/2 connection sharing feature, all `roUrlTransfers` should be made from the same thread.