Gmail **Awad Bin-Jawed <awadbinjawed@gmail.com>**

# GlideSystem Methods

1 message

**Awad Bin-Jawed** <awadbinjawed@gmail.com> Sun, Apr 14, 2019 at 3:02 PM
To: Awad Bin-Jawed <awadbinjawed@gmail.com>

## General

| Method Summary | Description |
|---|---|
| eventQueue(String, Object, String, String, String) | Queues an event for the event manager. |
| getCurrentScopeName() | Gets the name of the current scope. |
| getDisplayColumn(String) | Gets the display column for the table. |
| getDisplayValueFor(String, String, String) | Gets the display value for a given field. |
| getEscapedProperty(String, Object) | Gets the property and escapes it for XML parsing. |
| getMessage(String, Object) | Retrieves translated messages to display in the UI. If the specified string exists in the database for the current language, then the translated message is returned. If the specified string does not exist for the current language, then the English version of the string is returned. If the string does not exist at all in the database, then the ID itself is returned. |
| getMessageS(String, Object) | Retrieves translated messages to display in the UI and escapes all ticks ('). If the specified string exists in the database for the current language, then the translated message is returned. If the specified string does not exist for the current language, then the English version of the string is returned. If the string does not exist at all in the database, then the ID itself is returned. |
| getNodeValue(object, Integer) | Gets the node value for specified index. |
| getNodeName(Object, Integer) | Returns the node name for specified index. |
| getProperty(String, Object) | Gets the value of a Glide property. |
| getScriptError(String) | Returns the script error found in the specified script, if there is one. The script is not executed by this function, only checked for syntax errors. |
| getStyle(String, String, String) | Returns the style defined for the table, field and value. |
| getXMLText (String, String) | Gets the xml text for the first node in the xml string that matches the path query. |
| getXMLNodeList(String) | Constructs an Array of all the nodes and values in an XML document. |
| log(String message, String source) | Logs a message to the system log and saves it to the syslog table. |

| | |
|---|---|
| logError(String message, String source) | Logs an error to the system log and saves it to the syslog table. |
| logWarning(String message, String source) | Logs a warning to the system log and saves it to the syslog table. |
| nil(Object) | Queries an object and returns **true** if the object is null or contains an empty string. |
| print(String) | Writes a message to the system log. This method does not write the message to the syslog table unless debug has been activated. |
| tableExists(String) | Determines if a database table exists. |
| workflowFlush(Object) | Deletes all existing workflow operations for the specified GlideRecord. |

### Date and Time Functions

| **Method Summary** | **Description** |
|---|---|
| beginningOfLastMonth() | Gets the date and time for the beginning of last month in GMT. |
| beginningOfLastWeek() | Gets the date and time for the beginning of last week in GMT. |
| beginningOfNextWeek() | Gets the date and time for the beginning of next week in GMT. |
| beginningOfNextMonth() | Gets the date and time for the beginning of next month in GMT. |
| beginningOfNextYear() | Gets the date and time for the beginning of next year in GMT. |
| beginningOfThisMonth() | Gets the date and time for the beginning of this month in GMT. |
| beginningOfThisQuarter() | Gets the date and time for the beginning of this quarter in GMT. |
| beginningOfThisWeek() | Gets the date and time for the beginning of this week in GMT. |
| beginningOfThisYear() | Gets the date and time for the beginning of this week in GMT. |
| beginningOfToday() | Gets the date and time for the beginning of today in GMT. |
| beginningOfYesterday() | Gets the date and time for the beginning of yesterday in GMT. |
| calDateDiff(String, String, boolean) | Calculate the difference between two dates using the default calendar. **Note:** Calendars are now legacy. If Schedules are being used, see Calculate Duration Given a Schedule. |
| dateDiff(String, String, boolean) | Calculates the difference between two dates. The parameters must be in the user/system date time format. |
| dateGenerate(String, String) | Generates a date and time for the specified date in GMT. |
| daysAgo(int) | Gets a date and time for a certain number of days ago. The result is expressed in GMT. |
| daysAgoEnd(int) | Gets a date and time for end of the day a certain number of days ago.The result is expressed in GMT. |
| daysAgoStart(int) | Gets a date and time for beginning of the day a certain number of days ago. The result is expressed in GMT. |
| endOfLastMonth() | Gets the date and time for the end of last month in GMT. |
| endOfLastWeek() | Gets the date and time for the end of last week in GMT, in the format **yyyy-mm-dd hh:mm:ss.** |
| endOfLastYear() | Gets the date and time for the end of last year in GMT. |
| endOfNextMonth() | Gets the date and time for the end of next month in GMT. |
| endOfNextWeek() | Gets the date and time for the end of next week in GMT. |
| endOfNextYear() | Gets the date and time for the end of next year in GMT. |

| | |
|---|---|
| endOfThisMonth() | Gets the date and time for the end of this month in GMT. |
| endOfThisQuarter() | Gets the date and time for the end of this quarter in GMT. |
| endOfThisWeek() | Gets the date and time for the beginning of this week in GMT. |
| endOfThisYear() | Gets the date and time for the end of this year in GMT. |
| endOfToday() | Gets the date and time for the end of today in GMT. |
| endOfYesterday() | Gets the date and time for the end of yesterday in GMT. |
| hoursAgo(int) | Gets a date and time for a certain number of hours ago.The result is expressed in GMT. |
| hoursAgoEnd(int) | Gets a date and time for the end of the hour a certain number of hours ago.The result is expressed in GMT. |
| hoursAgoStart(int) | Gets a date and time for the start of the hour a certain number of hours ago.The result is expressed in GMT. |
| lastWeek() | Date and time one week ago in GMT. |
| minutesAgo(int) | Gets a date and time for a certain number of minutes ago.The result is expressed in GMT. |
| minutesAgoEnd(int) | Gets a date and time for the end of the minute a certain number of minutes ago.The result is expressed in GMT. |
| minutesAgoStart(int) | Gets a date and time for a certain number of minutes ago. The result is expressed in GMT. |
| monthsAgo(int) | Gets a date and time for a certain number of months ago.The result is expressed in GMT. |
| monthsAgoEnd(int) | Gets a date and time for the last day of the month a certain number of months ago.The result is expressed in GMT. |
| monthsAgoStart(int) | Gets a date and time for the first day of the month a certain number of months ago.The result is expressed in GMT. |
| now() | Gets the current date using GMT date time. |
| nowNoTZ() | Gets the current GMT date time. |
| nowDateTime() | Gets the current date and time in the user's time zone. |
| quartersAgo(int) | Gets a date and time for a certain number of quarters ago. The result is expressed in GMT. |
| quartersAgoEnd(int) | Gets a date and time for the last day of the quarter a certain number of quarters ago. The result is expressed in GMT. |
| quartersAgoStart(int) | Gets a date and time for the first day of the quarter a certain number of quarters ago. The result is expressed in GMT. |
| yearsAgo(int) | Gets a date and time for a certain number of years ago.The result is expressed in GMT. |
| yesterday() | Gets yesterday's time. The result is expressed in GMT. |
| isFirstDayOfMonth(Object) | Checks whether the date is the first day of the month. |
| isFirstDayOfWeek(Object) | Checks whether the date is the first day of the week. This uses the ISO standard of Monday being the first day of the week. |
| isFirstDayOfYear(Object) | Checks whether the date is the first day of the year |
| isLastDayOfMonth(Object) | Checks whether the date is the last day of the month. |
| isLastDayOfWeek(Object) | Checks whether the date is the last day of the week. |

| | |
|---|---|
| isLastDayOfYear(Object) | Checks whether the date is the last day of the year. |

### User Session Functions

| Method Summary | Description |
|---|---|
| addErrorMessage(Object) | Adds an error message for the current session. Session error messages are shown at the top of the form. Use getErrorMessages() to retrieve the list of error messages that are being shown. |
| addInfoMessage(Object) | Adds an info message for the current session. Session info messages are shown at the top of the form below any error messages. Use getInfoMessages() to retrieve the list of info messages being shown. |
| addMessage(String, Object) | Adds a message for the current session. Can be called using getMessages(String). |
| flushMessages() | Clears session messages saved using addErrorMessage(Object) or addInfoMessage(Object). Session messages are shown at the top of the form. In client side scripts use g_form.clearMessages() to clear all session messages. |
| getErrorMessages() | Gets the list of error messages for the session that were added by addErrorMessage(Object). |
| getImpersonatingUserDisplayName() | Returns the display name of the impersonating user. |
| getImpersonatingUserName() | Returns the name of the impersonating user or null if not impersonating. |
| getInfoMessages() | Gets the list of info messages for the session that were added via addInfoMessage(Object. |
| getMessages(String) | Gets the list of messages of the specified type for the session that were added via addMessage(String, Object). |
| getPreference(String, Object) | Gets a user preference. |
| getSession() | Returns a GlideSession object. |
| getSessionID() | Accesses the GlideSession Session ID. |
| getTrivialMessages() | Gets the list of error messages for the session that were added with the trivial flag. |
| getUser() | Returns a reference to the User object for the current user. More information is available here. |
| getUserDisplayName() | Returns the name field of the current user (e.g. John Smith, as opposed to smith). |
| getUserID() | Returns the sys_id of the current user. |
| getUserName() | Returns the username of the current user (for example, jsmith). |
| getUserNameByUserID(String) | Gets the username based on a user ID. |
| hasRole(String) | Determines if the current user has the specified role. |
| hasRoleInGroup(Object, Object) | Determines if the current user has the specified role within a specified group. |
| isInteractive() | Checks if the current session is interactive. |
| isLoggedIn() | Determines if the current user is currently logged in. |

| | |
|---|---|
| setRedirect(Object) | Sets the redirect URI for this transaction. This determines the next page the user will see. |
| setReturn(Object) | Sets the return URI for this transaction. This determines what page the user will be directed to when they return from the next form. |
| userID() | Returns the sys_id of the user associated with this session. A shortcut for the more proper getUserID(). |

# General

## eventQueue(String, Object, String, String, String)

Queues an event for the event manager.

### Input Fields

**Parameters:**

- Name of the event being queued.
- A GlideRecord object, such as "current".
- An optional parameter, saved with the instance if specified.
- A second optional parameter, saved with the instance if specified.
- An event queue to add the event to.

### Example

<source lang="javascript"> if (current.operation() != 'insert' && current.comments.changes()) {

```
   gs.eventQueue('incident.commented', current, gs.getUserID(), gs.getUserName());
```

} </source>

## getCurrentScopeName()

Gets the name of the current scope.

### Input Fields

**Parameters:**

- None

### Example

<source lang="javascript"> gs.getCurrentScopeName(); </source>

## getDisplayColumn(String)

Gets the display column for the table.

### Input Fields

**Parameters:** name of table from which to get the display column name.

### Output Fields

**Returns:** name of the display column for the table.

### Example

<source lang="javascript"> // Return the sys_id value for a given table and its display value function GetIDValue(table, displayValue) {

```
    var rec = new GlideRecord(table);
    var dn = gs.getDisplayColumn(table);
    if (rec.get(dn, displayValue))
        return rec.sys_id;
    else
        return null;
```

} </source>

## getDisplayValueFor(String, String, String)

Gets the display value for a specified field on a record.

### Input Fields

**Parameters:**

- String - name of table for the record
- String - the sysid for the record to get the display value from
- String - the field to get a display value from

### Output Fields

**Returns:** name of the display value for the field's value.

## getEscapedProperty(String, Object)

Gets the property and escapes it for XML parsing.

### Input Fields

**Parameters:**

- String key for the property whose value should be returned
- Alternate object to return if the property is not found.

### Output Fields

**Returns:** the property as a string, or the alternate object specified above.

## getMessage(String, Object)

Retrieves translated messages to display in the UI. If the specified string exists in the database for the current language, then the translated message is returned. If the specified string does not exist for the current language, then the English version of the string is returned. If the string does not exist at all in the database, then the ID itself is returned.

Note: if the UI message has a tick ('), there may be issues with the message in the script; to escape the ticks ('), use getMessageS(String, Object).

### Input Fields

**Parameters:**

- ID of message

- (Optional) A list of strings or other values defined by java.text.MessageFormat , which allows you to produce language-neutral messages for display to users.

### Output Fields

**Returns:** the UI message as a string.

### Examples

<source lang="javascript"> var my_message = '${gs.getMessage("This is a message.")}'; alert(my_message); </source>

Using the optional parameter as in: <source lang="javascript"> gs.getMessage('"{0}" is not a Client Callable Script Include','BAR’); </source>

Returns: "BAR" is not a Client Callable Script Include

## getMessageS(String, Object)

Retrieves translated messages to display in the UI and escapes all ticks ('). If the specified string exists in the database for the current language, then the translated message is returned. If the specified string does not exist for the current language, then the English version of the string is returned. If the string does not exist at all in the database, then the ID itself is returned. Useful if you are inserting into a JavaScript expression from Jelly.

For instance, the following code snippet will fail if the returned snippet has a tick ('):

<source lang="javascript"> var my_message = '${gs.getMessage("I love France")}'; alert(my_message); </source>

The solution is to use a snippet like the following:

<source lang="javascript">var my_message = '${gs.getMessageS("I love France")}'; alert(my_message); </source>

### Input Fields

**Parameters:**

- ID of message
- Optional message arguments.

### Output Fields

**Returns:** the message as a string, with the ticks escaped.

### Example

<source lang="javascript">var my_message = '${gs.getMessageS("I love France")}'; alert(my_message); </source>

## getProperty(String, Object)

Gets the value of a Glide property. If the property is not found, return an alternate value.
Use getEscapedProperty(String, Object) to escape the property.

### Input Fields

**Parameters**:

- String key for the property whose value should be returned.
- Alternate object to return if the property is not found.

### Output Fields

**Returns:** (String) the value of the Glide property, or the alternate object defined above.

### Example

<source lang="javascript">

```
//Check for attachments and add link if there are any
var attachment_link = ;
var rec = new GlideRecord('sc_req_item');
rec.addQuery('sys_id', current.request_item);
rec.query();
if(rec.next()){
  if(rec.hasAttachments()){
    attachment_link = gs.getProperty('glide.servlet.uri') + rec.getLink();
  }
}
```

</source>

## getScriptError(String)

Returns the script error found in the specified script, if there is one. The script is not executed by this function, only checked for syntax errors.

### Input Fields

**Parameters:** string script to check for errors.

### Output Fields

**Returns:** the script error message or, if none, then **null.**

## getStyle(String, String, String)

Returns the style defined for the table, field and value.

### Input Fields

**Parameters:**

- Table name
- Field name
- Field value

### Output Fields

**Returns** the style as a string.

## log(String message, String source)

Logs a message to the system log and saves it to the syslog table.

### Input Fields

**Parameters:**

- **String message** - message to log, for the log's **Message** field.
- **String source** - (optional) the source of the message, for the log's **Source** field.

### Example

<source lang="javascript"> var count = new GlideAggregate('incident'); count.addQuery('active', 'true'); count.addAggregate('COUNT', 'category'); count.query(); while (count.next()) {

```
   var category = count.category;
   var categoryCount = count.getAggregate('COUNT', 'category');
   gs.log("The are currently " + categoryCount + " incidents with a category of " +
 category, "Incident Counter");
```

} </source>

## logError(String message, String source)

Logs an error to the system log and saves it to the syslog table.

### Input Fields

**Parameters:**

- **String message** - message to log, for the log's **Message** field.
- **String source** - (optional) the source of the message, for the log's **Source** field.

## logWarning(String message, String source)

Logs a warning to the system log and saves it to the syslog table.

### Input Fields

**Parameters:**

- **String message** - message to log, for the log's **Message** field.
- **String source** - (optional) the source of the message, for the log's **Source** field.

## nil(Object)

Queries an object and returns **true** if the object is null or contains an empty string.

### Input Fields

**Parameters:** name of an object

### Output Fields

**Returns:** true if null or empty string; otherwise, returns false

### Example

<source lang="javascript"> if ((!current.u_date1.nil()) && (!current.u_date2.nil())) {

```
  var start = current.u_date1.getGlideObject().getNumericValue();
  var end = current.u_date2.getGlideObject().getNumericValue();
  if (start > end) {
    gs.addInfoMessage('start must be before end');
    current.u_date1.setError('start must be before end');
    current.setAbortAction(true);
  }
```

} </source>

## print(String)

Writes a message to the system log. This method does not write the message to the syslog table unless debug has been activated.

### Input Fields

**Parameters:** message to log

### Example

<source lang="javascript"> var rec = new GlideRecord('incident'); rec.addQuery('active',false); rec.query(); while (rec.next()) {

```
gs.print('Inactive incident ' + rec.number + ' deleted');
rec.deleteRecord();
```

} </source>

## tableExists(String)

Determines if a database table exists.

### Input Fields

**Parameters:** name of table to check for existence

### Output Fields

**Returns:** true if table exists or false is not found

## workflowFlush(Object)

Deletes all existing workflow operations for the specified GlideRecord.

### Input Fields

**Parameters:** the GlideRecord to flush the workflow for

### Output Fields

**Returns:** void

# Date/Time

## beginningOfLastMonth()

Gets the date and time for the beginning of last month in GMT.

### Output Fields

**Returns:** the GMT beginning of last month, in the format **yyyy-mm-dd hh:mm:ss.**

## beginningOfLastWeek()

Gets the date and time for the beginning of last week in GMT.

### Output Fields

**Returns:** the GMT beginning of last week, in the format **yyyy-mm-dd hh:mm:ss.**

## beginningOfNextWeek()

Gets the date and time for the beginning of next week in GMT.

### Output Fields

**Returns:** the GMT beginning of next week, in the format **yyyy-mm-dd hh:mm:ss.**

## beginningOfNextMonth()

Gets the date and time for the beginning of next month in GMT.

### Output Fields

**Returns:** the GMT beginning of next month, in the format **yyyy-mm-dd hh:mm:ss.**

## beginningOfNextYear()

Gets the date and time for the beginning of next year in GMT.

### Output Fields

**Returns:** the GMT beginning of next year, in the format **yyyy-mm-dd hh:mm:ss.**

## beginningOfThisMonth()

Gets the date and time for the beginning of this month in GMT.

### Output Fields

**Returns:** the GMT beginning of this month, in the format **yyyy-mm-dd hh:mm:ss.**

## beginningOfThisQuarter()

Gets the date and time for the beginning of this quarter in GMT.

### Output Fields

**Returns:** the GMT beginning of this quarter, in the format **yyyy-mm-dd hh:mm:ss.**

## beginningOfThisWeek()

Gets the date and time for the beginning of this week in GMT.

### Output Fields

**Returns:** the GMT beginning of this week, in the format **yyyy-mm-dd hh:mm:ss.**

## beginningOfThisYear()

Gets the date and time for the beginning of this week in GMT.

### Output Fields

**Returns:** the GMT beginning of this week, in the format **yyyy-mm-dd hh:mm:ss.**

## beginningOfToday()

Gets the date and time for the beginning of today in GMT.

### Output Fields

**Returns:** the GMT beginning of today, in the format **yyyy-mm-dd hh:mm:ss.**

## beginningOfYesterday()

Gets the date and time for the beginning of yesterday in GMT.

### Output Fields

**Returns:** the GMT beginning of yesterday, in the format **yyyy-mm-dd hh:mm:ss.**

## calDateDiff(String, String, boolean)

Calculate the difference between two dates using the default calendar. **Note:** Calendars are now legacy. If Schedules are being used, see Calculate Duration Given a Schedule.

### Input Fields

**Parameters:**

- **startDate** - a starting date to compare, in the current user's date format
- **endDate** - an ending date to compare, in the current user's date format
- **boolean numericValue** - if true, the return will be formatted in number of seconds; if false, the return will be formatted ddd hh:mm:ss.

### Output Fields

**Returns:** if the numericValue boolean parameter is true, returns the difference between the two dates as an integer number of seconds; if false, returns the difference between the two dates in the format ddd hh:mm:ss.

## dateDiff(String, String, boolean)

Calculates the difference between two dates. This method expects the earlier date as the first parameter and the later date as the second parameter; otherwise, the method returns the difference as a negative value. **Note:** Use getDisplayValue() to convert the strings to the expected format.

### Input Fields

**Parameters:**

- **startDate** - a starting date to compare, in the current user's date format.
- **endDate** - an ending date to compare, in the current user's date format.
- **boolean bnumericValue** - true to return difference in number of seconds as a string, false to return difference in the format ddd hh:mm:ss.

### Output Fields

**Returns:** if boolean bnumericValue is true, the difference in number of seconds; if false, the difference in the format ddd hh:mm:ss.

### Example

For more examples, see Setting the Duration Field Value.

<source lang="javascript"> // Given two date/times as DateTime objects // Set the values this way to ensure a consistent input time var date1 = new GlideDateTime(); var date2 = new GlideDateTime(); date1.setDisplayValueInternal('2014-01-01 12:00:00'); date2.setDisplayValueInternal('2014-01-01 13:00:00');

// Determine the difference as number of seconds (returns a string) // Use getDisplayValue() to convert the string to the format expected by dateDiff() var diffSeconds = gs.dateDiff(date1.getDisplayValue(), date2.getDisplayValue(), true);

// JavaScript will coerce diffSeconds from a string to a number // since diffSeconds is being compared to a number var msg = (diffSeconds <= 0) ? ' is on or after ' : ' is before '; gs.print(date1.getDisplayValue() + msg + date2.getDisplayValue()) </source>

## dateGenerate(String, String)

Generates a date and time for the specified date in GMT.

### Input Fields

**Parameters:**

- **Date** - format: yyyy-mm-dd
- **Range** - **start**, **end**, or a time in the 24 hour format **hh:mm:ss**.

### Output Fields

**Returns:** a date and time in the format yyyy-mm-dd hh:mm:ss. If range is **start**, the returned value is yyyy-mm-dd 00:00:00; If range is **end** the return value is yyyy-mm-dd 23:59:59.

## daysAgo(int)

Gets a date and time for a certain number of days ago.

### Input Fields

**Parameters:** An integer number of days ago.

### Output Fields

**Returns:** The (GMT) beginning of the days that was the specified number of days ago, in the format **yyyy-mm-dd hh:mm:ss.**

### Example

<source lang="javascript"> function contractNoticeDue() {

```
   var gr = new GlideRecord("contract");
   gr.addQuery("u_contract_status", "Active");
   gr.query();
   while (gr.next()) {
       if ((gr.u_termination_date <= gs.daysAgo(-90)) && (gr.u_contract_duration ==
"Long")) {
           gr.u_contract_status = "In review";
       }
       else if ((gr.u_termination_date <= gs.daysAgo(-50)) && (gr.u_contract_duration
== "Medium")) {
           gr.u_contract_status = "In review";
       }
       else if ((gr.u_termination_date <= gs.daysAgo(-10)) && (gr.u_contract_duration
== "Short")) {
           gr.u_contract_status = "In review";
       }
   }
   gr.update();
```

} </source>

## daysAgoEnd(int)

Gets a date and time for end of the day a certain number of days ago.

### Input Fields

**Parameters:** An integer number of days ago.

### Output Fields

**Returns:** The (GMT) end of the day that was the specified number of days ago, in the format **yyyy-mm-dd hh:mm:ss.**

## daysAgoStart(int)

Gets a date and time for beginning of the day a certain number of days ago.

### Input Fields

**Parameters:** An integer number of days ago.

### Output Fields

**Returns:** The (GMT) start of the day that was the specified number of days ago, in the format **yyyy-mm-dd hh:mm:ss.**

### Example

<source lang="javascript"> var gr = new GlideRecord('sysapproval_approver'); gr.addQuery('state', 'requested'); gr.addQuery('sys_updated_on', '<', gs.daysAgoStart(5)); gr.query(); </source>

## endOfLastMonth()

Gets the date and time for the end of last month in GMT.

### Output Fields

**Returns:** the GMT end of last month, in the format **yyyy-mm-dd hh:mm:ss.**

## endOfLastWeek()

Gets the date and time for the end of last week in GMT, in the format **yyyy-mm-dd hh:mm:ss.**

### Output Fields

**Returns:** the GMT end of last week, in the format **yyyy-mm-dd hh:mm:ss.**

## endOfLastYear()

Gets the date and time for the end of last year in GMT.

### Output Fields

**Returns:** the GMT end of last year, in the format **yyyy-mm-dd hh:mm:ss.**

## endOfNextMonth()

Gets the date and time for the end of next month in GMT.

### Output Fields

**Returns:** the GMT end of next month, in the format **yyyy-mm-dd hh:mm:ss.**

## endOfNextWeek()

Gets the date and time for the end of next week in GMT.

### Output Fields

**Returns:** the GMT end of next week, in the format **yyyy-mm-dd hh:mm:ss.**

## endOfNextYear()

Gets the date and time for the end of next year in GMT.

### Output Fields

**Returns:** the GMT end of next year, in the format **yyyy-mm-dd hh:mm:ss.**

## endOfThisMonth()

Gets the date and time for the end of this month in GMT.

### Output Fields

**Returns:** the GMT end of this month, in the format (yyyy-mm-dd huh:mm:ss)

## endOfThisQuarter()

Gets the date and time for the end of this quarter in GMT.

### Output Fields

**Returns:** the GMT end of this quarter, in the format **yyyy-mm-dd hh:mm:ss.**

## endOfThisWeek()

Gets the date and time for the beginning of this week in GMT.

### Output Fields

**Returns:** the GMT beginning of this week, in the format **yyyy-mm-dd hh:mm:ss.**

## endOfThisYear()

Gets the date and time for the end of this year in GMT.

### Output Fields

**Returns:** the GMT end of this year, in the format **yyyy-mm-dd hh:mm:ss.**

## endOfToday()

Gets the date and time for the end of today in GMT.

### Output Fields

**Returns:** the GMT end of today, in the format **yyyy-mm-dd hh:mm:ss.**

## endOfYesterday()

Gets the date and time for the end of yesterday in GMT.

### Output Fields

**Returns:** the GMT end of yesterday, in the format (yyyy-mm-dd huh:mm:ss).

## hoursAgo(int)

Gets a date and time for a certain number of hours ago.

### Input Fields

**Parameters:** An integer number of hours ago.

### Output Fields

**Returns:** The (GMT) time that was the specified number of hours ago,in the format **yyyy-mm-dd hh:mm:ss.**

### Example

<source lang="javascript"> if (current.operation() == 'insert') {

```
// If no due date was specified, calculate a default
if (current.due_date == ) {

 if (current.urgency == '1') {
  // Set due date to 4 hours ahead of current time
  current.due_date = gs.hoursAgo(-4);
 }

 if (current.urgency == '2') {
  // Set due date to 2 days ahead of current time
  current.due_date = gs.daysAgo(-2);
 }

 if (current.urgency == '3') {
  // Set due date to 7 days ahead of current time
  current.due_date = gs.daysAgo(-7);
 }
}
```

} </source>

## hoursAgoEnd(int)

Gets a date and time for the end of the hour a certain number of hours ago.

### Input Fields

**Parameters:** An integer number of hours ago.

### Output Fields

**Returns:** The (GMT) end of the hour that was the specified number of hours ago, in the format **yyyy-mm-dd hh:mm:ss.**

## hoursAgoStart(int)

Gets a date and time for the start of the hour a certain number of hours ago.

### Input Fields

**Parameters:** An integer number of hours ago.

### Output Fields

**Returns:** The (GMT) start of the hour that was the specified number of hours ago, in the format **yyyy-mm-dd hh:mm:ss.**

## lastWeek()

Date and time one week ago in GMT.

### Output Fields

**Returns:** the date and time one week ago, in the format **yyyy-mm-dd hh:mm:ss.**

## minutesAgo(int)

Gets a date and time for a certain number of minutes ago.

### Input Fields

**Parameters:** An integer number of minutes ago.

### Output Fields

**Returns:** The (GMT) time that was the specified number of minutes ago, in the format **yyyy-mm-dd hh:mm:ss.**

### Example

<source lang="javascript"> // // Check to see if the user has failed to login too many times // when the limit is reached, lock the user out of the system //

```
  //Check failed logins in the last 15 minutes
  var gr = new GlideRecord('sysevent');
  gr.addQuery('name', 'login.failed');
  gr.addQuery('parm1', event.parm1.toString());
  gr.addQuery('sys_created_on','>=', gs.minutesAgo(15));
  gr.query();
  var rowCount = gr.getRowCount();
  if(rowCount >= 5){
      var gr = new GlideRecord("sys_user");
      gr.addQuery("user_name", event.parm1.toString());
      gr.query();
      if (gr.next()) {
           gr.locked_out = true;
           gr.update();
           gs.log("User " + event.parm1 + " locked out due to too many invalid login
 attempts");
```

```
        }
    }
```

</source>

## minutesAgoEnd(int)

Gets a date and time for the end of the minute a certain number of minutes ago.

### Input Fields

**Parameters:** An integer number of minutes ago.

### Output Fields

**Returns:** The (GMT) end of the minute that was the specified number of minutes ago, in the format **yyyy-mm-dd hh:mm:ss.**

## minutesAgoStart(int)

Gets a date and time for the start of the minute a certain number of minutes ago.

### Input Fields

**Parameters:** An integer number of minutes ago.

### Output Fields

**Returns:** The (GMT) start of the minute that was the specified number of minutes ago, in the format **yyyy-mm-dd hh:mm:ss.**

## monthsAgo(int)

Gets a date and time for a certain number of months ago.

### Input Fields

**Parameters:** An integer number of months ago.

### Output Fields

**Returns:** A value (GMT) on the current (today's) date of the specified month, in the format **yyyy-mm-dd hh:mm:ss.**

## monthsAgoEnd(int)

Gets a date and time for the last day of the month a certain number of months ago.

### Input Fields

**Parameters:** An integer number of months ago.

### Output Fields

**Returns:** The (GMT) end of the month that was the specified number of months ago, in the format **yyyy-mm-dd hh:mm:ss.**

## monthsAgoStart(int)

Gets a date and time for the start of the minute a certain number of minutes ago.

### Input Fields

**Parameters:** An integer number of minutes ago.

### Output Fields

**Returns:** The (GMT) start of the minute that was the specified number of minutes ago, in the format **yyyy-mm-dd hh:mm:ss.**

## now()

Gets the current date using GMT.

### Output Fields

**Returns:** The current date in the user defined format, according to GMT.

### Example

<source lang="javascript"> // When the user password changes then set the u_password_last_reset field // to now so we know when to force another update

var gr = new GlideRecord("sys_user"); if (gr.get(event.parm1.toString())) {

```
    // Do something based on the Password Changing
    gs.log("The user password changed so do something else...");
    gr.u_password_last_reset = gs.now();
    gr.update();
```

}</source>

## nowNoTZ()

Gets the current date and time in UTC format.

### Output Fields

**Returns:** the current UTC date time.

### Example

<source lang="javascript"> // When the user password changes then set the u_password_last_reset field // to now so we know when to force another update

var gr = new GlideRecord("sys_user"); if (gr.get(event.parm1.toString())) {

```
    // Do something based on the Password Changing
    gs.log("The user password changed so do something else...");
    gr.u_password_last_reset = gs.nowNoTZ();
    gr.update();
```

}</source>

## nowDateTime()

Gets the current date and time in the user-defined format.

### Output Fields

**Returns:** The current date and time in the user-defined format.

### Example

The following script sets the field **u_target_date** to the current date and time:

<source lang="javascript"> current.u_target_date = gs.nowDateTime(); </source>

After the script is run, the **u_target_date** field will hold the date and time of the moment the script is run, in the system format.

### Example 2

When setting a variable in a workflow script to the current date and time, use the setDisplayValue method. The following script sets the workflow variable **end_date** to the current date and time:

<source lang="javascript"> current.variables.end_date.setDisplayValue(gs.nowDateTime()); </source>

After the script is run, the **end_date** field will hold the date and time of the moment the script is run, in the system format.

## quartersAgo(int)

Gets a date and time for a certain number of quarters ago.

### Input Fields

**Parameters:** An integer number of quarters ago.

### Output Fields

**Returns:** The (GMT) beginning of the quarter that was the specified number of quarters ago, in the format **yyyy-mm-dd hh:mm:ss.**

## quartersAgoEnd(int)

Gets a date and time for the last day of the quarter a certain number of quarters ago.

### Input Fields

**Parameters:** An integer number of quarters ago.

### Output Fields

**Returns:** The (GMT) end of the quarter that was the specified number of quarters ago, in the format **yyyy-mm-dd hh:mm:ss.**

## quartersAgoStart(int)

Gets a date and time for the first day of the quarter a certain number of quarters ago.

### Input Fields

**Parameters:** An integer number of quarters ago.

### Output Fields

**Returns:** The (GMT) end of the month that was the specified number of quarters ago, in the format **yyyy-mm-dd hh:mm:ss.**

## yearsAgo(int)

Gets a date and time for a certain number of years ago.

### Input Fields

**Parameters:** An integer number of years ago.

### Output Fields

**Returns:** The (GMT) beginning of the years that was the specified number of years ago, in the format **yyyy-mm-dd hh:mm:ss.**

## yesterday()

Gets yesterday's time.

### Output Fields

**Returns:** The (GMT) for 24 hours ago, in the format **yyyy-mm-dd hh:mm:ss.**

## isFirstDayOfMonth(Object)

Checks whether the date is the first day of the month.

### Input Fields

**Parameters:** date object.

### Output Fields

**Returns:** true if date is the first day of the month, false if it is not.

## isFirstDayOfWeek(Object)

Checks whether the date is the first day of the week. This uses the ISO standard of Monday being the first day of the week.

### Input Fields

**Parameters:** date object.

### Output Fields

**Returns:** true if date is the first day of the week, false if it is not.

## isFirstDayOfYear(Object)

Checks whether the date is the first day of the year.

### Input Fields

**Parameters:** date object.

### Output Fields

**Returns:** true if date is the first day of the year, false if it is not.

## isLastDayOfMonth(Object)

Checks whether the date is the last day of the month.

### Input Fields

**Parameters:** date object.

### Output Fields

**Returns:** true if date is the last day of the month, false if it is not. This uses the ISO standard of Sunday being the last day of the week.

## isLastDayOfWeek(Object)

Checks whether the date is the last day of the week.

### Input Fields

**Parameters:** date object.

### Output Fields

**Returns:** true if date is the last day of the week, false if it is not.

## isLastDayOfYear(Object)

Checks whether the date is the last day of the year.

### Input Fields

**Parameters:** date object.

### Output Fields

**Returns:** true if date is the last day of the year, false if it is not.

# User Session

## addErrorMessage(Object)

Adds an error message for the current session. Use getErrorMessages() to retrieve a list of error messages currently being shown.

### Input Fields

**Parameters:** an error object.

### Example

<source lang="javascript">

```
gs.include("PrototypeServer");
 var ValidatePasswordStronger = Class.create();
 ValidatePasswordStronger.prototype = {
      process : function() {
         var user_password = request.getParameter("user_password");
         var min_len = 8;
         var rules = "Password must be at least " + min_len +
            " characters long and contain a digit, an uppercase letter, and a lowercase
letter.";
```

```
        if (user_password.length() < min_len) {
           gs.addErrorMessage("TOO SHORT: " + rules);
           return false;
        }
        var digit_pattern = new RegExp("[0-9]", "g");
        if (!digit_pattern.test(user_password)) {
           gs.addErrorMessage("DIGIT MISSING: " + rules);
           return false;
        }
        var upper_pattern = new RegExp("[A-Z]", "g");
        if (!upper_pattern.test(user_password)) {
           gs.addErrorMessage("UPPERCASE MISSING: " + rules);
           return false;
        }
        var lower_pattern = new RegExp("[a-z]", "g");
        if (!lower_pattern.test(user_password)) {
           gs.addErrorMessage("LOWERCASE MISSING: " + rules);
           return false;
        }
        return true; // password is OK
     }
}
```

</source>

## addInfoMessage(Object)

Adds an info message for the current session. Use getInfoMessages() to retrieve the list of info messages being shown. **Note**: This method is not supported for asynchronous business rules and cannot be used within transform scripts.

### Input Fields

**Parameters:** an info message object.

### Example

<source lang="javascript"> if ((!current.u_date1.nil()) && (!current.u_date2.nil())) {

```
var start = current.u_date1.getGlideObject().getNumericValue();
var end = current.u_date2.getGlideObject().getNumericValue();
if (start > end) {
  gs.addInfoMessage('start must be before end');
  current.u_date1.setError('start must be before end');
  current.setAbortAction(true);
}
```

} </source>

## addMessage(String, Object)

Adds a message for the current session. Can be called using getMessages(String).

### Input Fields

**Parameters:**

- String type of message
- Message to store

### Example

<source lang="javascript"> gs.include("FormInfoHeader"); var fi = new FormInfoHeader(); var s = 'An incident ' + current.number + ' has been opened for your request.
'; s += 'The IT department will contact you when the password is reset or for further information.
'; //s += 'You can track status from the <a href="home.do" class="breadcrumb" >Home Page</a>
'; fi.addMessage(s); producer.redirect = 'home.do?sysparm_view=ess'; </source>

## flushMessages()

Clears session messages saved using addErrorMessage(Object) or addInfoMessage(Object). Session messages are shown at the top of the form. In client side scripts use g_form.clearMessages() to remove session messages.

### Output Fields

**Returns:** void

### Example

<source lang="javascript"> gs.flushMessages(); </source>

## getErrorMessages()

Gets the list of error messages for the session that were added by addErrorMessage(Object)

### Output Fields

**Returns:** list of error messages.

## getImpersonatingUserDisplayName()

Returns the display name of the impersonating user.

### Output Fields

**Returns:** display name of impersonating user.

## getImpersonatingUserName()

Returns the name of the impersonating user or null if not impersonating

### Output Fields

**Returns:** name of the impersonating user.

## getInfoMessages()

Gets the list of info messages for the session that were added via addInfoMessage(Object).

### Output Fields

**Returns:** the list of info messages.

## getMessages(String)

Gets the list of messages of the specified type for the session that were added via addMessage(String, Object).

### Input Fields

**Parameters:** string type of message.

### Output Fields

**Returns:** list of messages of string type.

## getNodeValue(object, Integer)

Gets the node value for specified index.

### Input Fields

**Parameters**

- Object to examine.
- Integer for the index to get a node value from.

### Output Fields

**Returns:** the node's value.

## getNodeName(Object, Integer)

Returns the node name for specified index.

### Input Fields

**Parameters**

- Object to examine.
- Integer for the index to get a node value from.

### Output Fields

**Returns:** the node's name.

## getPreference(String, Object)

Gets a user preference.

### Input Fields

**Parameters:**

- String key for the preference.
- Object default value

### Output Fields

**Returns:** a string value for the preference. If null, uses the default value specified above.

## getSession()

Returns a GlideSession object. See the GlideSession APIs wiki page for methods to use with the GlideSession object.

### Output Fields

**Returns:** a GlideSession object for the current session.

### Example

<source lang="javascript"> var session = gs.getSession(); </source>

## getSessionID()

Accesses the GlideSession Session ID.

### Output Fields

**Returns:** the Session ID.

## getTrivialMessages()

Gets the list of error messages for the session that were added with the trivial flag.

### Output Fields

**Returns:** the list of messages.

## getUser()

Returns a reference to the User object for the current user. More information is available here.

### Output Fields

**Returns:** a reference to a User object.

### Example

<source lang="javascript"> var myUserObject = gs.getUser() </source>

## getUserDisplayName()

Returns the name field of the current user (e.g. John Smith, as opposed to smith).

### Output Fields

**Returns:** the name field of the current user (e.g. John Smith, as opposed to jsmith).

### Example

<source lang="javascript">

```
   <g2:evaluate var="jvar_current_user" expression="gs.getUserDisplayName()"/>
    $[JS:jvar_current_user]
```

</source>

## getUserID()

Returns the sys_id of the current user.

### Output Fields

**Returns:** the sys_id of the current user.

### Example

<source lang="javascript"> if (current.operation() != ‘insert’ && current.comments.changes()) {

```
    gs.eventQueue(“incident.commented”, current, gs.getUserID(), gs.getUserName());
```

} </source>

## getUserName()

Returns the username of the current user (e.g., jsmith).

### Output Fields

**Returns:** the username of the current user (e.g., jsmith).

### Example

<source lang="javascript"> //Add a comment when closing

```
    current.comments = "Closed by " + gs.getUserName() + " at " + gs.nowDateTime();
    gs.addInfoMessage("Close comment added");
```

} </source>

## getUserNameByUserID(String)

Gets the username based on a user ID.

### Input Fields

**Parameters:** a sys_id as a string.

### Output Fields

**Returns:** the username.

## getXMLText (String, String)

Gets the xml text for the first node in the xml string that matches the xpath query.

### Input Fields

**Parameters:**

- XML to search within
- xpath query to match

### Output Fields

**Returns:** the XML node value as text.

## getXMLNodeList(String)

Constructs an Array of all the nodes and values in an XML document.

### Input Fields

**Parameters:** XML document to parse.

### Output Fields

**Returns:** an array list of names and values.

## hasRole(String)

Determines if the current user has the specified role. This method returns true for users with the administrator role.

### Input Fields

**Parameters:** the role to check.

### Output Fields

**Returns:** true if the user has the role or false if the user does not have the role.

### Example

<source lang="javascript"> if (!gs.hasRole("admin") && !gs.hasRole("groups_admin") && gs.getSession().isInteractive()) {

```
  var qc = current.addQuery("u_hidden", "!=", "true"); //cannot see hidden groups...
  qc.addOrCondition("sys_id", "javascript:getMyGroups()"); //...unless in the hidden
group
```

} </source>

## hasRoleInGroup(Object, Object)

Determines if the current user has the specified role within a specified group.

### Input Fields

**Parameters:**

- The name of the role to check for.
- A GlideRecord or the sys_id of the group to check within.

### Output Fields

**Returns:** Returns true if all of the following conditions are met:

1. The logged-in user HAS the role in question
2. The "Granted by" field on the user role record is set to the specified group
3. The "inherited" field on the user role record is false

### Example

<source lang="javascript"> var group = new GlideRecord('sys_user_group'); group.addQuery('name', 'GROUP_NAME'); group.setLimit(1); group.query(); if (group.next()) {

```
   if (gs.hasRoleInGroup('role_name', group)) {
      gs.print('User has role in group');
   } else {
      gs.print('User does NOT have role in group');
   }
```

} </source>

# isInteractive()

Checks if the current session is interactive. An example of an interactive session is when a user logs in using the log-in screen. An example of a non-interactive session is using a SOAP request to retrieve data.

### Output Fields

**Returns:** true if the session is interactive.

### Example

<source lang="javascript"> if (!gs.hasRole('admin') && gs.isInteractive()) {

```
    var qc1 = current.addQuery('u_group', );
    var gra = new GlideRecord('sys_user_grmember');
    gra.addQuery('user', gs.getUserID());
    gra.query();
    while (gra.next()) {
      qc1.addOrCondition('u_group', gra.group);
  }
```

} </source>

# isLoggedIn()

Determines if the current user is currently logged in.

### Output Fields

**Returns:** true if the current user is logged in, false if the current user is not.

# setRedirect(Object)

Sets the redirect URI for this transaction. This determines the next page the user will see.

### Input Fields

**Parameters:** URI object to set as redirect.

### Example

The following example redirects the user to a particular Catalog Item, and passes along the current email as a parameter:

<source lang="javascript"> gs.setRedirect("com.glideapp.servicecatalog_cat_item_view.do?sysparm_id=d41ce5bac611227a0167f4bf8109bf70&sysparm_user=" + current.sys_id + "&sysparm_email=" + current.email) </source>

# setReturn(Object)

Sets the return URI for this transaction. This determines what page the user will be directed to when they return from the next form.

### Input Fields

**Parameters:** URI object to set as return.

### Example

The following example ensures that the user will be returned to the current page when they are done with the next one.

<source lang="javascript">

```
gs.setReturn (current.getLink(true));
```

</source>

## userID()

Returns the sys_id of the user associated with this session. A shortcut for the more proper getUserID().

### Output Fields

**Returns:** sys_id of current user.