# spUtil API Reference

**Service Portal Cheat Sheet**

brought to you by NEWROCKET

spUtil enables you to perform common functions in a Service Portal widget client script.

**How to Use**: Pass in spUtil as a parameter in your Client Script function.

```
function (spUtil) {
   spUtil.addInfoMessage("Hello World");
}
```

| Method (Parameters) | Description |
|---|---|
| **addErrorMessage**<br>(String message) | Displays a notification error message.<br>**Return**: (Void) |
| **addInfoMessage**<br>(String message) | Displays a notification info message.<br>**Return**: (Void) |
| **addTrivialMessage**<br>(String message) | Displays a trivial notification message. Trivial messages disappear after a short period of time.<br>**Return**: (Void) |
| **clearMessages**<br>(N/A) | Clears any info or error message. Broadcasts "$$uiNotification.dismiss" event to rootscope. |
| **createUid**<br>(String str) | Uses a provided string to generate a UID.<br>**Return**: (String) |
| **format**<br>(String template, Object data) | Formats a string as an alternative to string concatenation. Use this method to build a string with variables. This method returns a formatted string.<br>**Return**: (String) |
| **get**<br>(String widgetId, Object data) | Returns a widget model by ID or sys_id. Use this method to embed a widget model in a widget client script. The callback function returns the full widget model. Checks the specified list of field names, and returns an array of valid field names. This method returns the model of the embedded widget.<br>**Return**: (Object) |
| **getAccelerator**<br>(String char) | If Mac user agent, returns back "Cmd + char", else "Ctrl + char".<br>**Return**: (String) |
| **getHeaders**<br>(N/A) | Returns an object containing header details, includes an attribute which references the portal ID.<br>**Return**: (Object) |
| **getHost**<br>(N/A) | Returns base instance URL (including https:// prefix).<br>**Return**: (String) |
| **getMomentTimeZone**<br>(N/A) | This will attempt to map a three letter timezone symbol into it's fully qualified name.<br>**Return**: (String) |
| **getPageUri**<br>(N/A) | Returns page URI.<br>**Return**: (String) |
| **getPreference**<br>(String preferenceName, Function callback) | Looks up user preference value and then passes value into callback function.<br>**Return**: (Void) |

| Method (Parameters) | Description |
|---|---|
| **getURL**<br>(String type, Object type) | Utilized client side in conjunction with $http to submit catalog items.<br>**Return**: (String) |
| **getWidgetURL**<br>(N/A) | Returns SNOW API REST endpoint URL used to communicate with SNOW instance.<br>**Return**: (String) |
| **isMobile**<br>(N/A) | Scans current navigator user agent to determine if current user is on a mobile device.<br>**Return**: (Boolean) |
| **parseAttributes**<br>(String attributes) | Given a string of URL attributes, returns an object containing same detail.<br>**Return**: (String) |
| **recordWatch**<br>(Object $scope, String table, String filter, Function callback) | Watches for updates to a table or filter and returns the value from the callback function. Allows a widget developer to respond to table updates in real-time. This method returns the value of the callback function. When the record watcher triggers, it will call the callback function provided in this method (or simply trigger a $scope.server.update()).<br>**Return**: (Void) |
| **refresh**<br>(Object $scope) | Calls the server and replaces the current options and data with the server response. Calling spUtil.refresh() is similar to calling server.refresh(). However, when you call spUtil.refresh(), you can define the $scope object. This method returns the upated options and data objects.<br>**Return**: (Object) |
| **scrollTo**<br>(String elementID, Integer scrollDelayInMS) | Scrolls view to the selector provided in the time provided. The elementID refers to the actual ID of the element within the widget. After a given delay, the selected element will scroll into view.<br>**Return**: (Void) |
| **setBreadCrumb**<br>(Scope $scope, Array list) | Given a list of breadcrumb labels/values, update breadcrumbs on the form.<br>**Return**: (Void) |
| **setPreference**<br>(String preferenceName, String value) | Updates the user preference value.<br>**Return**: (Void) |
| **setSearchPage**<br>(String searchPage) | Should emit an update.searchpage event with the search page provided.<br>**Return**: (Void) |
| **update**<br>(String preferenceName, String value) | Updates the data object on the server within a given scope. This method is similar to server.update(), but includes a $scope parameter that defines the scope to pass over. This method returns the updated data object.<br>**Return**: (Object) |