Cheatography

ServicePortal Cheat Sheet Cheat Sheet

by Bibin Gokuldas (bibingokuldas) via cheatography.com/69673/cs/23975/

CSS Sequence (Execution Order)

» ServiceNow Specific StyleSheet
› Generated Bootstrap CSS
✱ Including Portal and Theme records' CSS at the top of this blob
» Patch
» Each CSS Include's Style Sheet record's CSS
✱ Including Portal and Theme records' CSS at the top of each blob
» Font-Awesome
» Page record's CSS
✱ Including Portal and Theme records' CSS at the top of this blob
» Alternating Sequence of:
» Each Widget record's CSS
✱ Including Portal and Theme record CSS at the top of each blob
» Each Widget's Instance records' Stylesheets
✱ Including Portal and Theme record CSS at the top of each blob

Angular Directives

Evaluate the given expression when the user changes the input.	`ng-change="expression"`
Sets the checked attribute on the element	`ng-checked="boolean"`
This directive sets the disabled attribute on the element	`ng-disabled="boolean"`
The ngHide directive shows or hides the given HTML element based on the expression provided to the ngHide attribute.	`ng-hide|show="boolean"`
The ngMousedown directive allows you to specify custom behavior on mousedown event.	`ng-mousedown="expression"`
Specify custom behavior on mouseover event	`ng-mouseover="expression"`
Specify custom behavior on blur event. A blur event fires when an element has lost focus.	`ng-blur="expression"`
Run a loop with the data in array	`ng-repeat="([key,] value) in object|array"`
Evaluate an if condition	`ng-if="expression"`

Client script global functions

`this.server.get([Object])`	Calls the server and sends custom
`this.server.update()`	Calls the server and posts this.data Promise.
`this.server.refresh()`	Calls the server and automatically r data from the server response. Ret

A promise represents the eventual result of an asynchror information on promises, see https://promisesaplus.com/

spAriaUtil - Client

Announce a message to a screen reader.	`spAriaUtil.sendLiv screen reader');`

The spAriaUtil service is an angular service included as p angular application. The spAriaUtil service is available in Portal widgets.

To use the spAriaUtil please add the same to client c

```
function(spAriaUtil) {
/ widget controller /
var c =this;
}
```

spUtil - Client

Displays a notification error message.	`spUtil.addErrorMes error processing y`
Displays a notification info message.	`spUtil.addInfoMess been placed")`
Displays a trivial notification message. *Trivial messages disappear after a short period of time.*	`spUtil.addTrivialN order")`

spUtil - Client (cont)

Use this method to embed a widget model in a widget client script. The callback function returns the full widget model.	`spUtil.get('pps-list-modal', {title: c.data.editAllocat-ions, table: 'resource_allocation', queryString: 'GROUPBYuser^re-source_plan=' + c.data.sysId, view: 'resource_portal_allo-cations' }).then(function(r-esponse) { var formModal = response; c.allocationListModal = response; });`
Formats a string as an alternative to string concatenation. *Use this method to build a string with variables.*	`spUtil.format('An error ocurred: {error} when loading {widget}', {error: '404', widget: 'sp-widget'})`
Calls the server and replaces the current options and data with the server response.	`spUtil.refresh($scope)`
Updates the data object on the server within a given scope.	`spUtil.update($scope)`
Watches for updates to a table or filter and returns the value from the callback function.	`spUtil.recordWatch($scope, "-problem", "active=true", function(response) { // Returns the data inserted or updated on the table console.log(response.data); });`

Utility methods to perform common functions in a Service Portal widget client script.

To use the spUtil functions in the client controller, ensure that you add to client function

```
function(spUtil,$scope) {
/ widget controller /
var c =this;
}
```

Internationalize a widget

✱Use the ${} or gs.getMessage() syntax in widgets to tag strings localize your Service Portal content.

```
<div>
<p>${This message will be translated.}</p>
<p>However, this will nessage will NOT be transla
</div>
```

Now to translate the above, add the text in sys_ui_message to get ex:

- Key: This message will be translated.
- Language : 'de' / 'fr' / 'it'
- Message : Add the translated text

GlideSPScriptable - Scoped

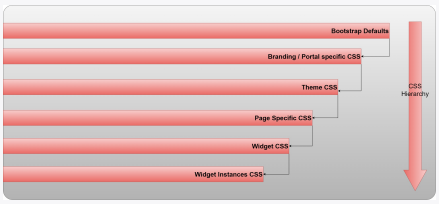
`if the user can read the specified GlideR-ecord.`	`$sp.c ord>)`
`if the user can read the specified record with table and sysID`	`$sp.c 'sys_`
`if the currently logged in user has permission to view the specified page`	`Glide e("pa`
`Returns a model and view model for a sc_cat_item or sc_cat_item_guide.`	`$sp.g`
`Returns the display value of the specified field (if it exists and has a value) from either the widget's sp_ins-tance or the sp_portal record.`	`$sp.g atalc`
`Returns the portal record from the Service Portals [sp_portal] table.`	`$sp.g`
`Returns an array of Service Catalog variables associated with a record.`	`$sp.g ('sc_ recor varia`

GlideSPScriptable - Scoped (cont)

Gets a widget by id or sys_id, executes that widget's server script using the provided options, then returns the widget model.	`$sp.getWidget('widget-_id', {p1: param1, p2: param2});`

Interact with data and perform record operations in Service Portal widgets. You access GlideSPScriptable methods by using the global $sp object.

CSS Hierarchy



Global objects in widgets

» **Server Side Objects**

› ***input*** :*Sent from Client script*

› ***options*** :*Configurations mentioned in widget options when widget is loaded*

› ***data*** :*Object or properties/value pair sent to client*

» **Client Side Objects**

› ***options*** :*Configurations mentioned in widget options when widget is loaded*

› ***data*** : *Serialized data object from Server script*

The "snRecordPicker"

```
✱ Creating a reference Field
<sn-record-picker field=""
                  table="'sys_user'"
                  display-field="'name'"
                  display-fields="'user_name'"
                  value-field="'sys_id'"
                  search-fields="'name'"
                  default-query="'nameLIKEAdmin'"
                  page-size="100" ></sn-record-picker>
```

When you try and include a reference field (such as "department") in the "display-fields" parameter, the field that you specify doesn't render other non reference fields do render correctly. This is an existing issue can be found in the KB0717097

you can pass in the filter you need to apply on reference field using default-query attribute

The "sp-date-picker"

```
✱ The sp-date-picker allo
time in widgets
<sp-date-picker field="fro
                ng-model="
                ng-model-o
true}"
                sn-include
                sn-disable
cker>
```

sn-include-time property when set t and time. and when false it will allo

FormController -(Widgets)

True if user has not interacted with

True if user has already interacted

spModal - Client

Displays an alert.	`spModal.a update') { c.simple });`
Displays a confirmation message.	`spModal.( or deny confirme( c.confir or false })`
Opens a modal window using the specified options.	`spModal.( title: '( message: name it? input: t: value: c }).then( c.name = })`

spModal - Client (cont)

Displays a prompt for user input.	`spModal.prompt("Your name please", c.name).then-(function(name) { c.name = name; })`

The spModal class provides an alternative way to show alerts, prompts, and confirmation dialogs. The SPModal class is available in Service Portal client scripts.
You can use spModal.open() to display a widget in a modal dialog.
To use spModal in the client controller ensure that you add the spModal function in the client function

```
function(spModal) {
/ widget controller /
var c =this;
}
```

Embed a widget in an HTML template

✱ Use the <widget></widget> element to embed a widget in an HTML template. Pass in the ID of the widget you are trying to embed as a parameter.

```
<div>
<widget id="widget-cool-clock"></widget>
</div>
```

✱If a widget has an option schema, you can define instance options in JSON format.

```
<widget id="widget-cool-clock" options='{"zone": "America/Los_Angele-s","title": "San Diego, CA"}'></widget>
```

spContextManager - Client

Initializes a key and adds widget data as the value.	`spContextManager.addContext('agent-chat', { 'appro-val_count': 5 });`
Returns each key and associated data object defined by any widget on the page.	`spContextManager.getContext();`
Returns the widget data associated with a key.	`spContextManager.getContextForKey('agent-chat', true).then(function(context) { context = context || {}; context.approval_count = 5; spContextManager.u-pdateContextForKey('agent-chat', context); });`

spContextManager - Client (cont)

Sends data to an existing key.	`spContextManager.getContextF true).then(function(context) {}; context.approval_count = pdateContextForKey('agent-ch`

Make data from a Service Portal widget available services in a Service Portal page.
to use the spContextManager, ensure to add controller

```
function ($scope, spContextManager)
var c =this;
}
```