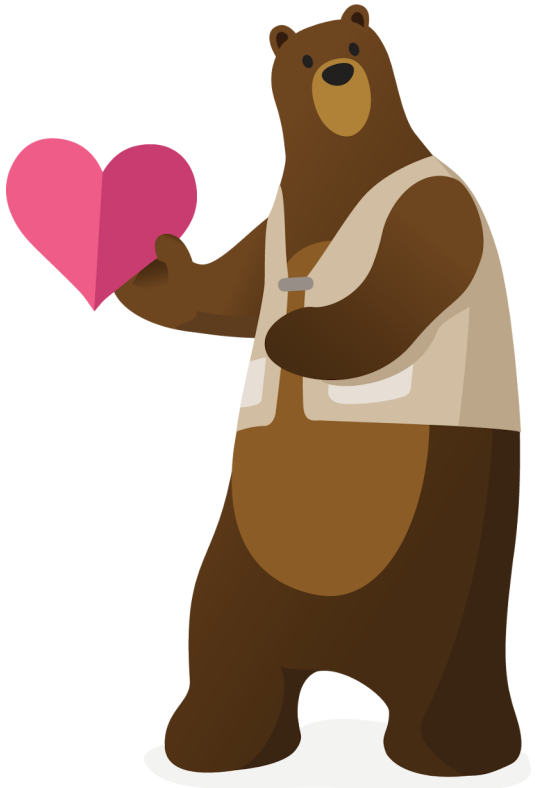
Chatter Answers JavaScript API Reference

Salesforce, Spring '21



'21

@salesforcedocs

Last updated: November 12, 2020

CONTENTS

CHATTER ANSWERS JAVASCRIPT API REFERENCE

The Chatter Answers JavaScript API helps you mediate communication to the Chatter Answers UI. Learn the standard events and methods that you can use to customize the behavior of Chatter Answers UI components.

Note: The Chatter Answers JavaScript API is currently available through a pilot program. For information on enabling Chatter Answers JavaScript API for your organization, contact salesforce.com.

By using the Chatter Answers JavaScript API, you can listen to events that are fired in the application. You can also register your own listeners for those events.

IN THIS SECTION:

Chatter Answers JavaScript API Events

The Chatter Answers JavaScript API has four events.

Note: The Chatter Answers JavaScript API is currently available through a pilot program. For information on enabling Chatter Answers JavaScript API for your organization, contact salesforce.com.

Use these events to modify the behavior of Chatter Answers UI components.

IN THIS SECTION:

HOVER_USER

The HOVER_USER event mediates communication to the Chatter Answers reputation hover UI module.

For example, when users hover over a user profile picture, the HOVER_USER event is triggered and you can register a callback to listen to this event.

Payload

Payload object	Description
userId	ID of the user
communityId	Zone ID of the feed item hovered over.
targetId	Reference to the target DOM element hovered over.

Example

```
sforce.answers.events.type.HOVER_USER
```

`publish()` Example for HOVER_USER

```
var $answers = sforce.answers;
$answers.events.publish($answers.events.type.HOVER_USER);
```

`subscribe()` Example for HOVER_USER

```
var $answers = sforce.answers;
var myHandler = function(evtData) {
     alert("Now I am also listening to the HOVER_USER event");
};
$answers.events.subscribe($answers.events.type.HOVER_USER, myHandler, this);
```

FEED_READY

The FEED_READY event indicates that the Chatter Answers feed completed loading.

This event is fired by Chatter Answers. You should trigger other custom events only after FEED_READY is fired. This event cannot be overridden.

Payload

None

Example

```
var myHandler = function() {
alert("The Feed is ready");
}
$answers.events.subscribe($answers.events.type.FEED_READY, myHandler);
```

FEED_ITEM_SELECTED

Triggered when a search result from the Search/Ask component is selected.

This event causes the feed to display the selected item in single item view. FEED_ITEM_SELECT can be published to cause the feed to display a feed item in single item view. At least one Chatter Answers component or the `<allfeeds>` component must be present on the Chatter Answers page for this event to fire. FEED_ITEM_SELECT can be subscribed to, in order to register additional callbacks to the event. The default handler of the event cannot be overridden.

Payload

ID of the feed item to be rendered in single item view.

`publish()` Example for FEED_ITEM_SELECTED

```
var $answers = sforce.answers;
$answers.events.publish($answers.events.type.FEED_ITEM_SELECTED, {id:<feed_item_id>});
```

`subscribe()` Example for FEED_ITEM_SELECTED

```
var myHandler = function(evtPayload) {
alert("Feed item with id = "+ evtPayload.id +" was selected!");
}
$answers.events.subscribe($answers.events.type.FEED_ITEM_SELECTED, myHandler);
```

EDIT_MY_SETTINGS_CLICK

Triggered by default when the Edit My Settings link is clicked.

The default handler causes the My Settings overlay to appear. The EDIT_MY_SETTINGS event can be published to cause the My Settings overlay to display. The overlay appears only if a user is logged in and at least one Chatter Answers component is on the page. This event can be subscribed to, allowing a registered callback. However, this event doesn't override the default behavior of the My Settings overlay.

Payload

None

`publish()` Example for EDIT_MY_SETTINGS_CLICK

```
var $answers = sforce.answers;
$answers.events.publish($answers.events.type.EDIT_MY_SETTINGS_CLICK);
```

`subscribe()` Example forEDIT_MY_SETTINGS_CLICK

```
var myHandler = function() {
alert("My settings link was clicked!");
}
$answers.events.subscribe($answers.events.type.EDIT_MY_SETTINGS_CLICK, myHandler);
```

Chatter Answers JavaScript API Methods

The Chatter Answers JavaScript API has two methods.

Note: The Chatter Answers JavaScript API is currently available through a pilot program. For information on enabling Chatter Answers JavaScript API for your organization, contact salesforce.com.

`force.answers.events` is the namespace for the Chatter Answers JavaScript API methods. `type` refers to an event in a method.

Use these methods for the Chatter Answers Javascript API events.

IN THIS SECTION:

`publish()`

Use `publish()` to fire an event.

Parameters

Parameter	Description
eventType	Event type being fired.

Example

```
var $answers = sforce.answers;
$answers.events.publish($answers.events.type.HOVER_USER);
```

`subscribe()`

Use `subscribe()` to subscribe to an event.

Parameters

Parameter	Description
eventType	Event type to listen to.
handler	Handler function to execute when this event is fired.
context	Optional. Context in which the above handler function should be executed. Values:

	• `null` • `this` • some context
override	Optional. Override the default handler for this event. Values: • `true` • `false`

Examples

Example 1: To register my handler for an event.

```
var $answers = sforce.answers;
var myHandler = function(evtData) {
     alert("Now I am also listening to the HOVER_USER event");
};
$answers.events.subscribe($answers.events.type.HOVER_USER, myHandler, this);
```

or

```
$answers.events.subscribe($answers.events.type.HOVER_USER, myHandler);
var $answers = sforce.answers;
$answers.events.publish($answers.events.type.HOVER_USER);
```

Example 2: To register a handler for an event by overriding the default behavior for that event.

```
var $answers = sforce.answers;
var myOverridenHandler = function(evtData) {
       console.log("User id: " + evtData.userId);
       console.log("Community id: " + evtData.communityId);
       console.log("Target Element Id: " + evtData.targetId);
};
$answers.events.subscribe($answers.events.type.HOVER_USER, myOverridenHandler, null, true);
```