

QuickSight Q Embedding Developer Guide

Prerequisites

- AWS account with QuickSight Q enabled
- At least one Topic set up with QuickSight Q (and that topic's topic id)
- QuickSight session embedding framework setup with AWS account
 - Workshop for this: <https://learnquicksight.workshop.aws/en/session-embedding.html>
 - Be sure to allow list your domain in the Manage QuickSight page

Determining Topic(s) To Show

QuickSight Q embedding supports two different use cases regarding topics. The first is when a single topic is to be specified, and only that topic can be queried against via the search bar. The second is the standard experience within the QuickSight application, where a user has a list of topics and can use a dropdown in the search bar to select the topic that they would like to query against. Before proceeding, make sure you know whether your use case requires one topic, or a list of topics in the embedded experience.

Allowlisting Domain

Per the QuickSight embedding guides, you'll notice you need to allowlist your application's domain in the 'Manage QuickSight' page. Normally, this is all you need to do, but in the case of Q we'll also need to add the QuickSight domain to the allowlist. This seems weird, but since under the hood Q is also using an iframe, we need, at least for now, to allowlist the 'same' domain. The QuickSight domain you need to allowlist is dependent on the region you are using. For example, embedding using the 'us-east-1' region, our URL to allowlist would be:

```
https://us-east-1.quicksight.aws.amazon.com
```

The URL would be the same for other regions, with the respective regional part (us-east-1) changed.

Generating the new session URL

First the part of the embedding framework that generates the session URL via the getSessionEmbedURL API needs to be slightly modified. QuickSight Q session embedding is supported at a different 'entry point' than the standard console embedded experience. Documentation for this API can be [found here](#). The `entry-point` parameter for this API call will need to be changed - the new entry point is dependent on topic(s) use case.

For the single topic case:

```
entry-point = /q/search/<topicId>
```

To show all topics in the selector, we'll omit the `topicId`:

```
entry-point = /q/search
```

This should generate the one-time authenticated URL that will render a page with just the QuickSight Q search bar.

Using the JS SDK to Embed

With the URL, we can use the provided QuickSight Embedding Javascript SDK to embed the Q search bar in the application. First, make sure you have a [copy of the SDK](#) from the QuickSight team - note since this is a preview feature and is not released yet, the SDK will be a different version than the publicly available [SDK on github](#). We'll want to use the EmbedSession method from the SDK with the generated session URL. The relevant options for the embedSession function are (also found in types.js in the SDK):

```
url: url of the session or dashboard to embed
container: parent html element or query selector string
errorCallback: callback when error occurs
loadCallback: callback when visualization data load complete
className: optional className to be given to iframe element
isQEmbedded: embeddable object is Q search bar flag
maxHeightForQ: height for Q to resize to when it expands
onQBarOpenCallback: optional callback for Q search bar open
onQBarCloseCallback: optional callback for Q search bar close
```

The two required arguments here are `url` and `container`. We'll use the URL generated from the getSessionEmbedURL API call, and for the container this is dependent on your application. You'll want at the very least a simple `<div>` as the 'container' for the embedded iframe; give this container an `id` and pass in the `id` in the SDK arguments. The default session embedding callbacks, `errorCallback` and `loadCallback` do as the name might suggest - if you need custom behavior when the embedded page loads, or encounters an error, specify that logic in these callbacks. When using Q embedded mode with the SDK, the iframe will be a fixed height (the height of the search bar itself) and 100% width of the parent HTML container. This means that the search bar will only be as wide as the container; you'll want to make sure the search bar has at least 600px of width (whether translated from view-width/percentage or directly assigned). For styling the iframe, the `className` parameter can be optionally specified as well.

IMPORTANT:

A key callout here is to make sure that the `<div>` or component that you are passing as the container html element has a styling of `'position: absolute'`. This is what allows the search bar to expand as an overlay instead of shifting the contents of your app down.

QuickSight Q Embedding Changes

There are a few key differences between session/dashboard embedding and Q search bar embedding (although currently Q embedding simply utilizes session embedding). With dashboard and session embedding, the frame is generally a single size, barring some resizing based on the size of a dashboard or analysis sheet. With Q, initially the embedded frame on your page is relatively small (we only want the actual search bar to show). When the search bar is being used, this frame needs to expand (to show additional dropdown elements like the visual result, suggestions, etc). To expand this frame without shifting the contents of your application, we just set it as an overlay over the existing page, similar to how the search bar functions in the QuickSight application today - see screenshots below.



With a little about how Q embedding works in mind, let's look at the QuickSight Q specific SDK parameters. First, we'll need `isQEmbedded` to be set as `true`. `maxHeightForQ` is an optional argument that specifies the largest that the Q frame can be on your page; as mentioned previously, we'll need the `iframe` and it's container to expand over the contents of the page. We can use the `maxHeightForQ` argument to ensure the frame/container don't resize past your page's max height and cause a scroll to appear or other unwanted behavior. If not set, the Q frame will resize to `100vh`.

The last two Q specific parameters are callbacks that happen when the embedded frame resizes. The default behavior is to create a backdrop element, and use this to give the darker background appearance that we see in the above screenshots from the QuickSight application. This is the 'out of the box' functionality that we want to provide to make embedding Q as easy as possible - however, we know this won't work for every application that Q needs to be embedded in. If you have a need to override this behavior, simply write the logic into the `onQBarOpenCallback` and `onQBarCloseCallback`. This will prevent the default backdrop from showing up as well.

QuickSight Q Styling Options

There are a few styling/cosmetic options that we can use to customize the appearance of the Q search bar.

```
qBarIconDisabled: option to hide the Q search bar
qBarTopicNameDisabled: option to hide the Q search bar topic name
themeId: option to apply Quicksight theme to Q search bar
```

If you would like to disable the 'Q' icon (in the left hand side of the search bar, use the `qBarIconDisabled` parameter. Similarly, to disable the topic name, if you are only displaying a singular topic in embedded mode, use the `qBarTopicNameDisabled` parameter. Note that these cosmetic customizations are *only available for the case when you're embedding a single topic*.

If you want to theme the embedded Q bar to make the appearance consistent with your application, you can do so by creating a new theme in Quicksight and pass the `themeId` to SDK (example below).

Examples

The following examples will assume there is a container present in DOM with the id 'q-bar-container'.

Embedding With Default Backdrop Behavior (with q/search path)

Let's assume the container has a margin top of 75px, so we'll account for that using the `maxHeightForQ` parameter, so that the iframe does not expand larger than the page allows, creating a scroll bar or other unwanted behavior.

```
function embedQSession(embedUrl) {
  var containerDiv = document.getElementById("q-bar-container");
  containerDiv.innerHTML = "";
  var params = {
    url: embedUrl,
    container: containerDiv,
    isQEmbedded: true,
    maxHeightForQ: "calc(100vh - 75px)",
  };
  QuickSightEmbedding.embedSession(params);
}
```

Embedding With Disabled Backdrop Behavior (with q/search path)

For this example we'll assume the container is at the top of the page so it can expand to 100% without issue; we won't need the `maxHeightForQ`. We'll use the `onQBarOpenCallback` and the `onQBarCloseCallback` as no-op functions to prevent the default backdrop from appearing.

```
function embedQSession(embedUrl) {
  var containerDiv = document.getElementById("q-bar-container");
  containerDiv.innerHTML = "";
  var params = {
    url: embedUrl,
    container: containerDiv,
    isQEmbedded: true,
    onQBarOpenCallback: () => {},
    onQBarCloseCallback: () => {},
  };
  QuickSightEmbedding.embedSession(params);
}
```

Embedding With Custom Backdrop Behavior (with q/search path)

We'll again assume the container is at the top of the page so it can expand to 100% without issue; we won't need the `maxHeightForQ`. We'll use the `onQBarOpenCallback` and the `onQBarCloseCallback` as callbacks which manipulate some other backdrop component (`customBackdropComponent`) in our application that we'd like to use in place of the default one. Note your backdrop callbacks might be more complex, this example is just for simplicity.

```
function onQBarOpen() {
    customBackdropComponent.style.height = "100%";
}

function onQBarClose() {
    customBackdropComponent.style.height = 0;
}

function embedQSession(embedUrl) {
    var containerDiv = document.getElementById("q-bar-container");
    containerDiv.innerHTML = "";
    var params = {
        url: embedUrl,
        container: containerDiv,
        isQEmbedded: true,
        onQBarOpenCallback: () => {},
        onQBarCloseCallback: () => {},
    };
    QuickSightEmbedding.embedSession(params);
}
```

Embedding With Default Backdrop Behavior (with q/search/topicId path)

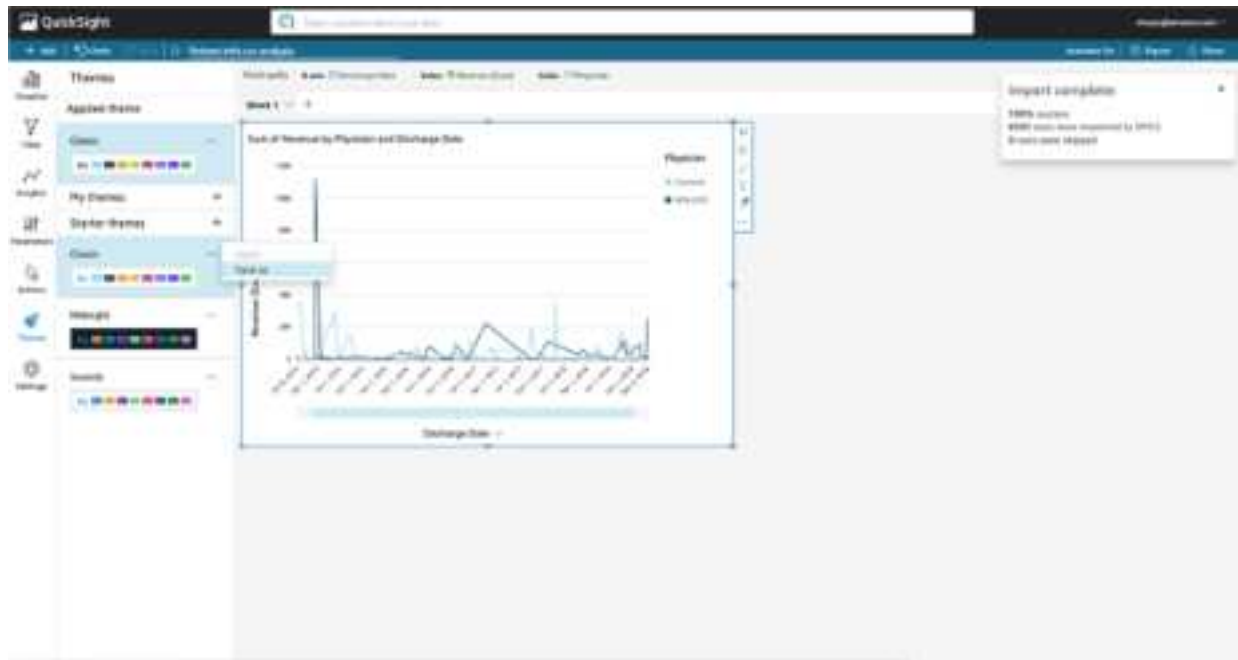
We'll again assume the container has a margin top of 75px, so we'll account for that using the `maxHeightForQ` parameter, so that the `iframe` does not expand larger than the page allows, creating a scroll bar or other unwanted behavior. Since we're using the embedded search bar with a single topic, we can use the `qBarIconDisabled` and `qBarTopicNameDisabled` customizations. This example will give us a search bar with no icon or topic name, set to be ready to query on whichever `topicId` is passed in.

```
function embedQSession(embedUrl) {
    var containerDiv = document.getElementById("q-bar-container");
    containerDiv.innerHTML = "";
    var params = {
        url: embedUrl,
        container: containerDiv,
        isQEmbedded: true,
        maxHeightForQ: "calc(100vh - 75px)",
        qBarIconDisabled: false,
        qBarTopicNameDisabled: false,
    };
    QuickSightEmbedding.embedSession(params);
}
```

Embedding With Theme Id

Create a new theme inside QuickSight if you don't have one. Open an analysis, or create a new one. Choose Themes at left.

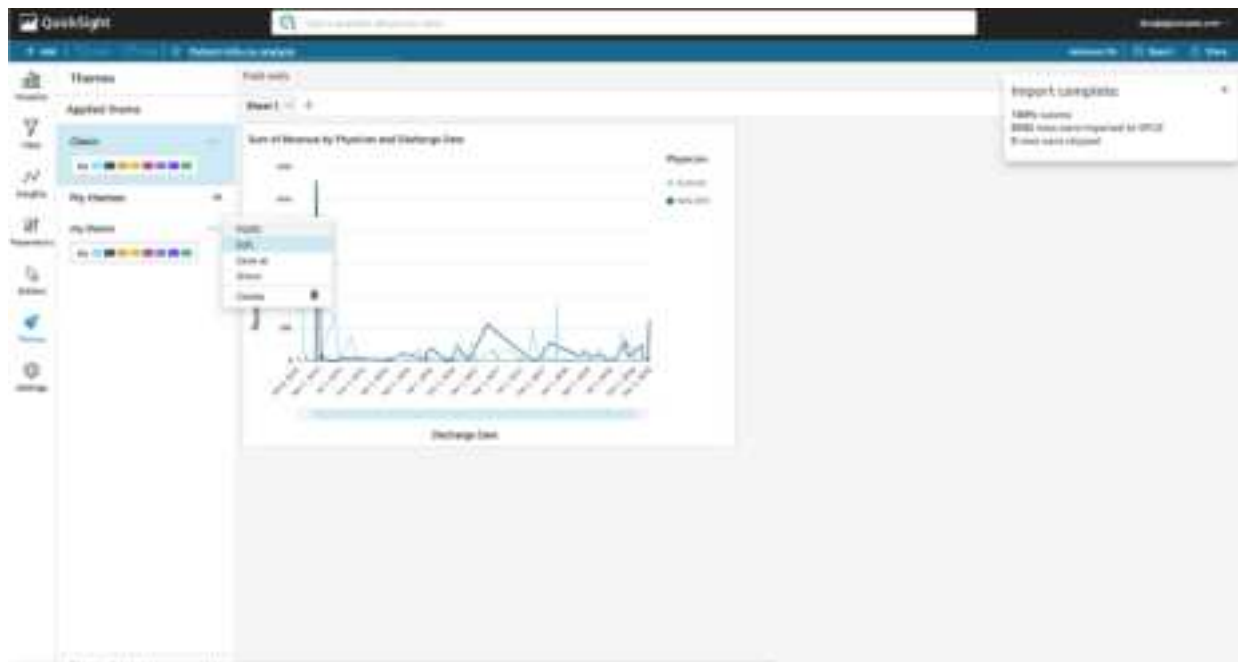
And then choose one of the starter theme you prefer then click “Save as”. If you already have a theme, you may skip the theme creation step.



It will take you to the theme editor page, give it a name, adjust the color as you like then save it on the top right.



Now that you have a theme, you need to find the id of that theme and pass it to SDK. Choose “Edit” on the theme you created.



It will again take you to theme editor page, but this time, you will find theme id there in the url bar. In this case “d39c0065-bf69-4b3d-927b-9dd3a241f094” is the id of a theme I created.







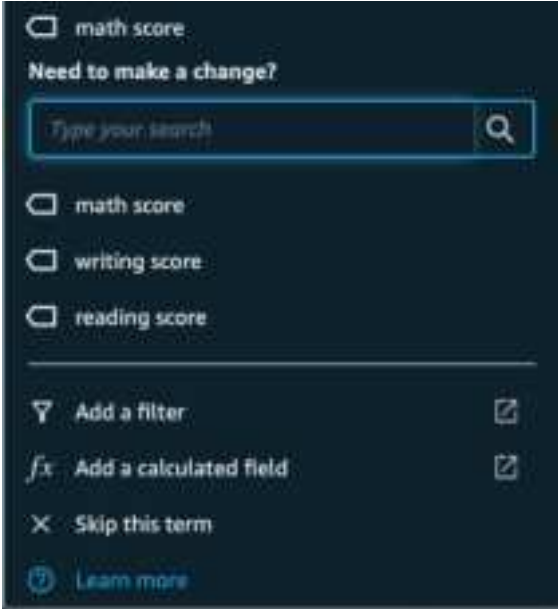

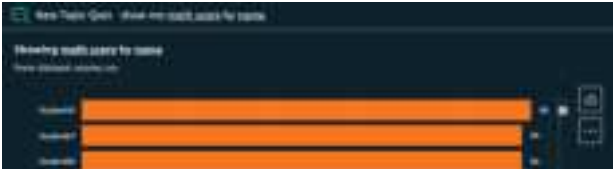
Lastly, you pass the theme id as a parameter to the SDK, then you will get a themed Q bar in you app.


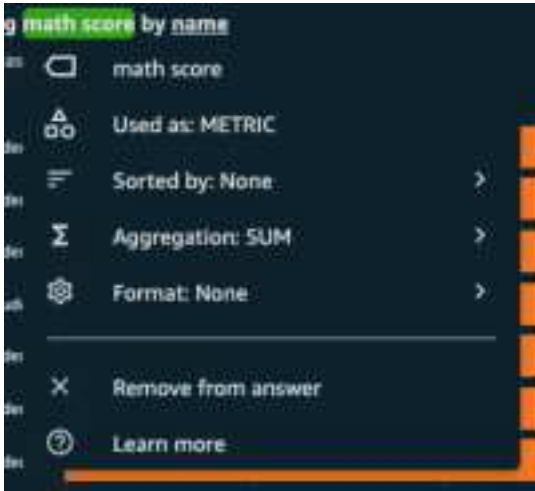
```
function embedQSession(embedUrl) {
  var containerDiv = document.getElementById("q-bar-container");
  containerDiv.innerHTML = "";
  var params = {
    url: embedUrl,
    container: containerDiv,
    isQEmbedded: true,
    maxHeightForQ: "calc(100vh - 75px)",
    qBarIconDisabled: false,
    qBarTopicNameDisabled: false,
    themeId: "d39c0065-bf69-4b3d-927b-9dd3a241f094"
  };
  QuickSightEmbedding.embedSession(params);
}
```



Themeable Component

We want to show you which components inside Q bar can be themed, and we will take QuickSight Midnight theme as an example (you can find it in starter themes)

Q component	Screenshot	Theming option
-------------	------------	----------------

Search bar background color		Primary background
Search bar placeholder text "Type a question about your data"		Secondary foreground
User question		Primary foreground
Visual title (restatement)		Primary foreground
Text in disambiguation dropdown		Primary foreground
Recent Question dropdown		Primary foreground
Top and bottom dividers		Primary foreground

		
Border for the hover over on the restatement entity		Primary foreground
Selected chart type		Accent
		Headers --> Primary foreground Text / description --> Secondary foreground

Feedback modal		Selection box --> primary foreground Selection box highlight --> Accent Background color --> primary background Text box border --> primary foreground text box highlight --> accent
Powered by quicksight		secondary foreground

Troubleshooting

'Refused to frame *.quicksight.aws.amazon.com because an ancestor violates..' Error

This error is caused by not allow listing your domain in the manage QuickSight admin page in the QuickSight application. Be sure to allow list both the regional QuickSight domain (for us-east-1, this is <https://us-east-1.quicksight.aws.amazon.com>, for example), as well as your application's domain.

Cannot See Search Bar After Successfully Embedding

If you are able to successfully generate and access and embedding link (with no permissions or other obvious errors), but get a 'blank' screen with no search bar appearing, there's a few things to check. One is that there must be at least one topic shared with the user you are embedding with. Second, you'll want to make sure there is at least one topic that is successfully created and in a 'successful' state after initially building. An easy way to test this is to just use the topic in Q in the QuickSight application; if it works normally, it's good to use in embedded mode.

Another reason the search bar may not be appearing is that the container the embedding iframe is in may not be giving enough width. As called out in the documentation, you'll want to make sure the search bar is given *at least* 600px of width for it to function normally.

Search Bar Expands But Shifts Content Down

The search bar with the SDK was designed to expand over top of any additional content on the page. If this is not the case, please make sure the search bar container <div> is styled with a 'position: absolute', which will allow it to not shift the contents of the page down. For example:

```
<div id="q-search-container" style="position: absolute; width: 100%"></div>
```

SDK Revision Change Log

- V1(5/15/21): Initial custom 'Q-ready' SDK
 - V1.1 (5/25/21)
 - Change iframe height to be fixed to the height of the search bar itself, user cannot specify in Q mode
 - Set iframe width to 100% of the parent container for Q mode. Note that the width of the search bar can still be limited by the size of the parent container.