VCS

Add text animations to Daily live streams with Daily's VCS baseline composition

Display dynamic live stream viewer feedback using Daily's new Video Component System (VCS) default compositions



Jess Mitchell

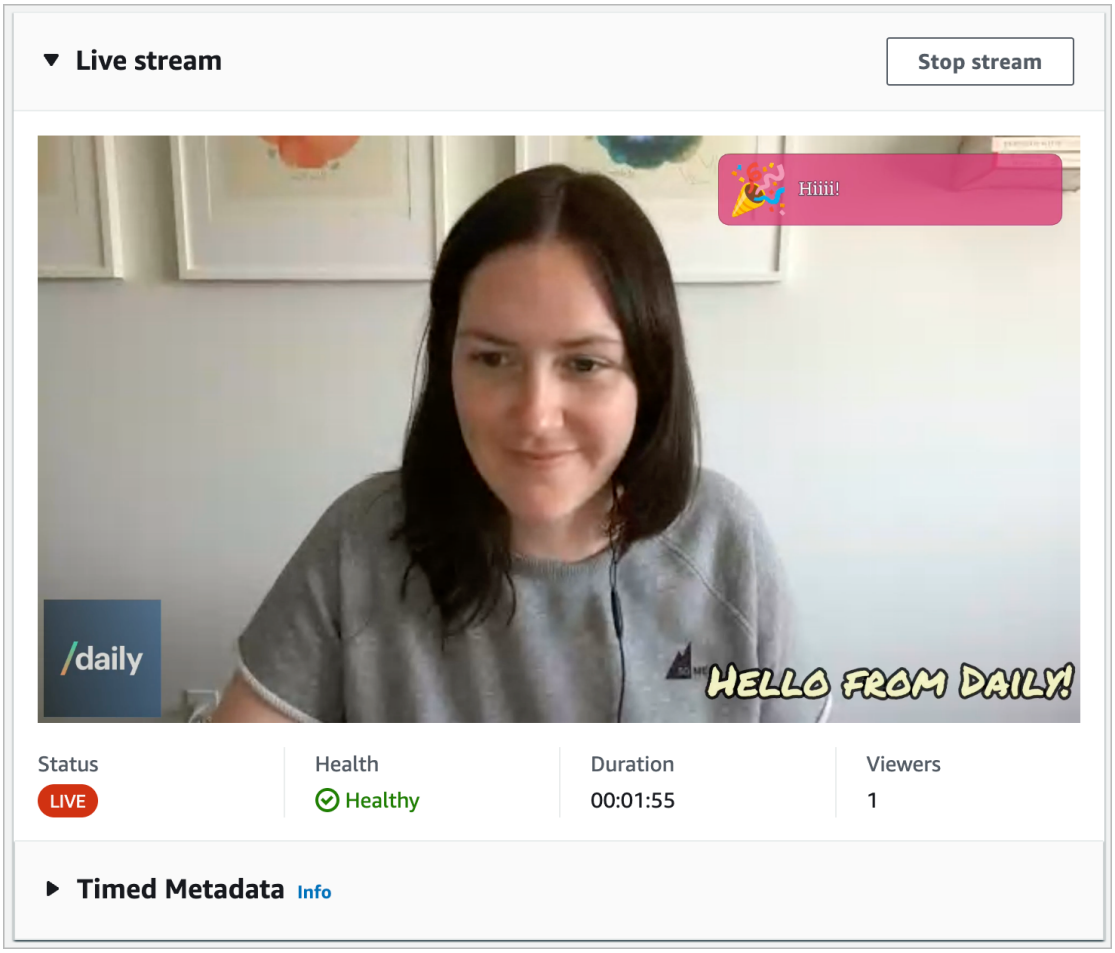
Published: 4 Jul 2022 (Updated: 4 Jul 2022) • 6 min read



In a recent live streaming tutorial, we looked at how to use Daily's Video Component System (VCS)'s baseline composition to create custom, multi-camera live stream layouts, as well as add image and text overlays that can be triggered mid-stream.

In today's tutorial, we'll extend this example and show how to add text animations (also known as "toasts") that can be displayed during a live

stream with dynamic text content. We recommend reading that first post for more thorough instructions on getting the demo app set up locally, as well as specific instructions on how to start a live stream. If you already know how to do that, keep reading!



Toast component displayed in the top right corner of a Daily live stream video

Note: These VCS features we've been demonstrating with live streams can be applied to recordings, as well. Read our VCS baseline composition guide for more information.

Before moving on to code examples, let's do a quick recap on Daily definitions:

- **VCS** (or Video Component System) is Daily's video-oriented layout engine that can be used to create layouts (or "compositions"), which turn Daily WebRTC calls into highly customized live streams or recorded content.
- The **baseline composition** is a sub-category of VCS. It is the default collection of layouts and graphic features accessible via the `daily-js` instance methods options for live streaming and recording.

- Keep an eye out for an exciting announcement soon that will give developers even more ways to customize live stream and recorded video layouts and graphics.

This tutorial will cover how to use VCS's baseline composition to display toast animations via the `updateLiveStreaming()` `daily-js` method. However, you could also do this through:

- `startLiveStreaming()`
- `startRecording()`
- `updateRecording()`

Additionally, you can also access the baseline composition through `react-native-daily-js` .

Psst... We have an exciting announcement related to VCS later this week. Keep an eye out on Twitter for more information.

Now that we know what we're using today, let's get started.

Running the `prebuilt-ui` demo locally

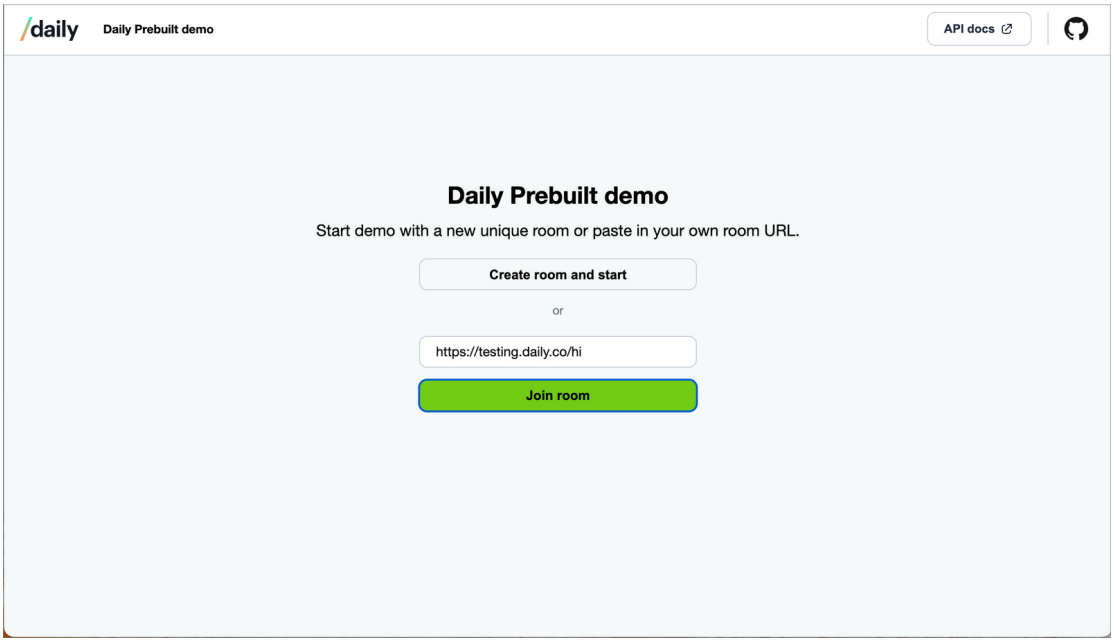
This tutorial will use the `vcs-live-streaming` branch of the `prebuilt-ui` demo app.

To run this branch locally, enter the following commands in your terminal:

```
git clone https://github.com/daily-demos/prebuilt-ui.git
cd prebuilt-ui
git checkout vcs-live-streaming
npm i
npm run dev
```

These commands will clone the repo, checkout the specific branch we're looking at today, install the dependencies, and then start your server.

Open the demo in your browser at `localhost:8080` and you should see the demo home screen.



Daily Prebuilt demo home screen

Next, you will need to add a `owner` meeting token and RTMP URL to `index.js` in the demo. These steps were explained in the previous post on using Daily's baseline composition, so give that a read for additional information for getting set up locally.

In our previous post, we looked at how to start live streaming in a Daily call with the baseline composition enabled, update our live stream layout mid-stream to use picture-in-picture (PiP), and add a text and image overlay.

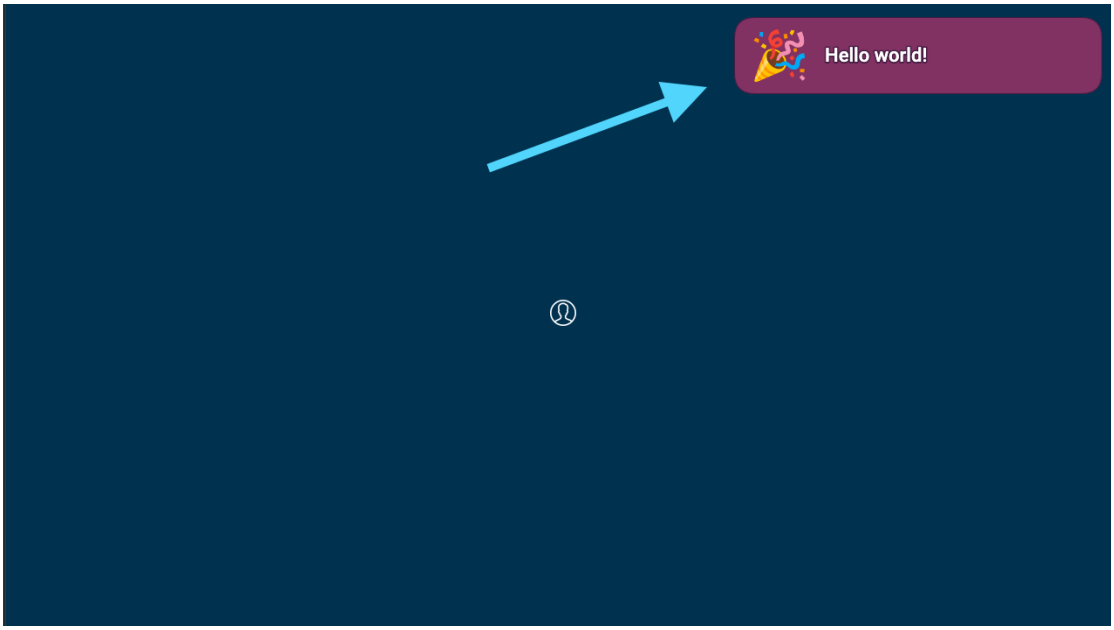
In today's post, we'll start where we left off with the live stream already started and those visual updates applied. All of these features can be used individually or together, though — it's up to you!

To learn more and test out all the features the baseline composition has to offer, we suggest visiting the VCS Simulator, which allows you to test all the features Daily VCS's baseline composition has to offer.

Adding a toast component to your live stream

In case it's not totally clear, let's quickly review what a toast component is exactly. It’s a message that pops up on the screen and then fades out after a few seconds.

In some cases, they're just emojis with no text (think of Instagram live reactions that float across the screen.)



Toast component triggered in VCS Simulator

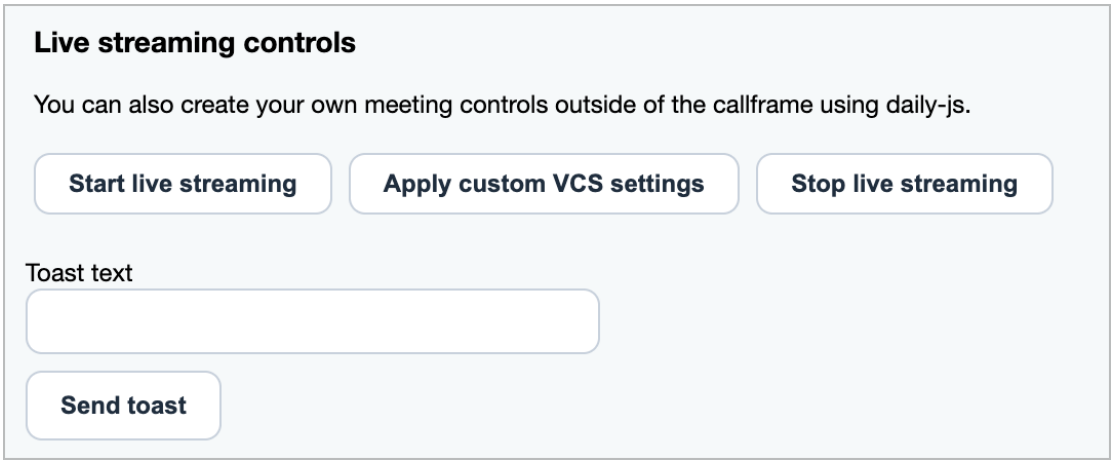
Toasts are a great way for viewers to passively give feedback or add input to a call without creating a disruption for the speaker.

Note: With VCS, the toast must be triggered through the `updateLiveStreaming()` `daily-js` ***(or*** `react-native-daily-js` ***) method call. To let live stream viewers control the toast content, you***

method call. To let live stream viewers control the toast content, you could, for example, have an endpoint in your app that sends the reaction to the live streamer so they can trigger the reaction via `updateLiveStreaming()`.

How to trigger a dynamic toast component

First, let's add a form to our control panel where the user can enter text in an input that will show in a toast component, and a button to submit the form.



Demo control panel with a toast input added

In the `index.html` file, it will look like this:

```
<form onsubmit="showToast(event)">
    <label for="toast">Toast text</label>
    <input class="toast-input" type="text" id="toast" />
    <div>
        <input
           class="white-button" type="submit"
           value="Send toast"
           />
    </div>
</form>
```

Our submit form has an `onsubmit` event handler that will call `showToast()`.

Now, let's define `showToast()`.

```
let toastKey = 0;

function showToast(e) {
 // prevent default form behavior
 e.preventDefault();
 // get input value from form submit event
 const toastText = e.target[0].value;

 // send toast
 callFrame.updateLiveStreaming({
   layout: {
     preset: 'custom',
     composition_params: {
       'toast.text': toastText,
       'toast.color': 'rgba(215, 50, 110, 0.8)',
       'toast.text.fontFamily': 'Bitter',
       'toast.duration_secs': 3,
       'toast.text.fontSize_pct': 150,
       'toast.text.fontWeight': '400',

       'toast.key': toastKey,
     },
   },
 });
 toastKey++;
}
```

Note: As a reminder, we're assuming `startLiveStreaming()` has already been called with the `layout.preset` value set to `custom` to enable access to the baseline composition.

Let's step through this:

1. First, we call `e.preventDefault()` to prevent the form submission from refreshing the page.
2. Next, we get the toast text from the form input.
3. Then, we call `updateLiveStreaming()`. (More on this in a second.)
4. Finally, we increment our `toastKey` value by 1, which is initially defined outside of the `showToast()` function.

Step three is where we're actually triggering the toast component.

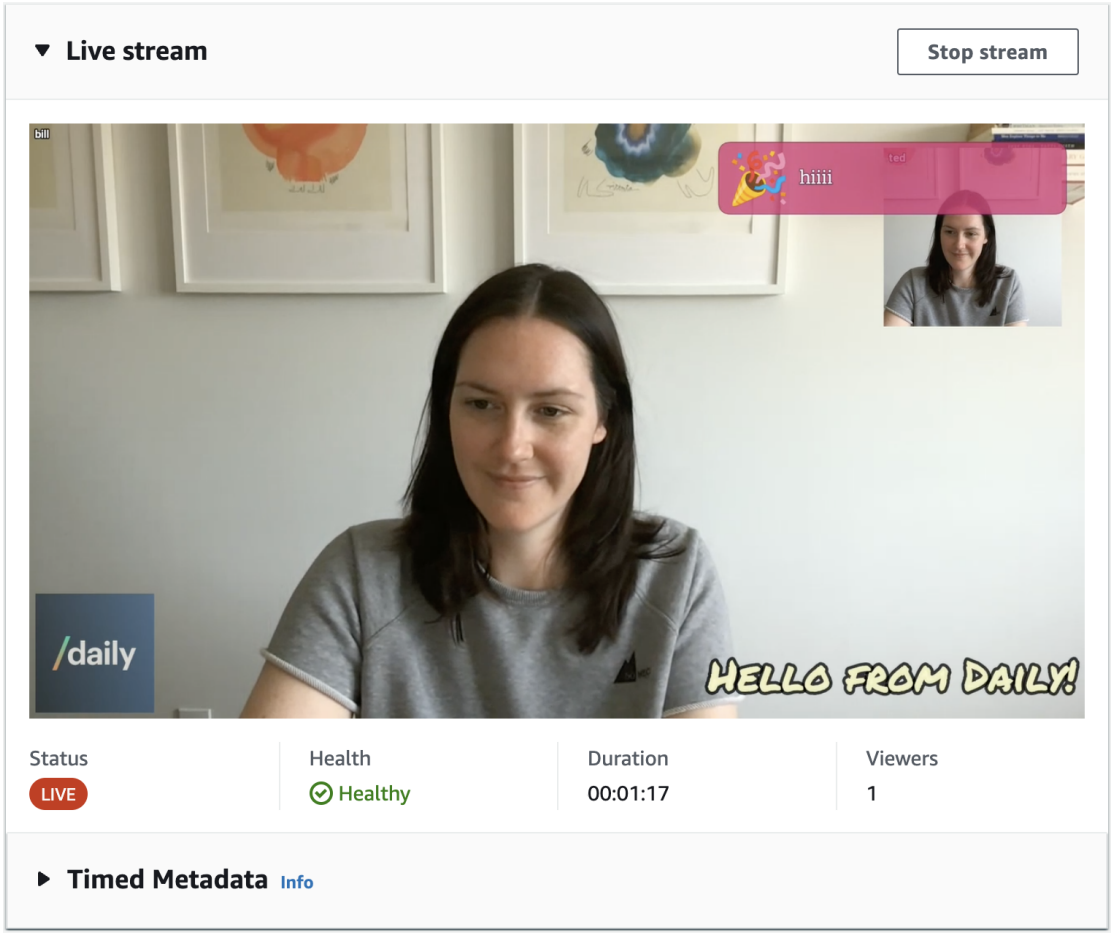
In `composition_params`, we're setting our toast text to use the form's input value:

```
'toast.text': toastText,
```

We're also updating the toast's style with some extra options (e.g. `'toast.text.fontWeight': '400'`)

The most important step is setting the `toast.key` value. It is initialized at `0` and needs to be incremented with each new toast to actually trigger that a new toast should be shown in the feed. That is why we do step four – increment our `toastKey` so it's ready for the next time a toast needs to be sent!

In the end, the toast will look like this:



Live stream with toast component temporarily shown

Once the toast is triggered, it will be shown to everyone viewing the live

Once the toast is triggered, it will be shown to everyone viewing the live stream.

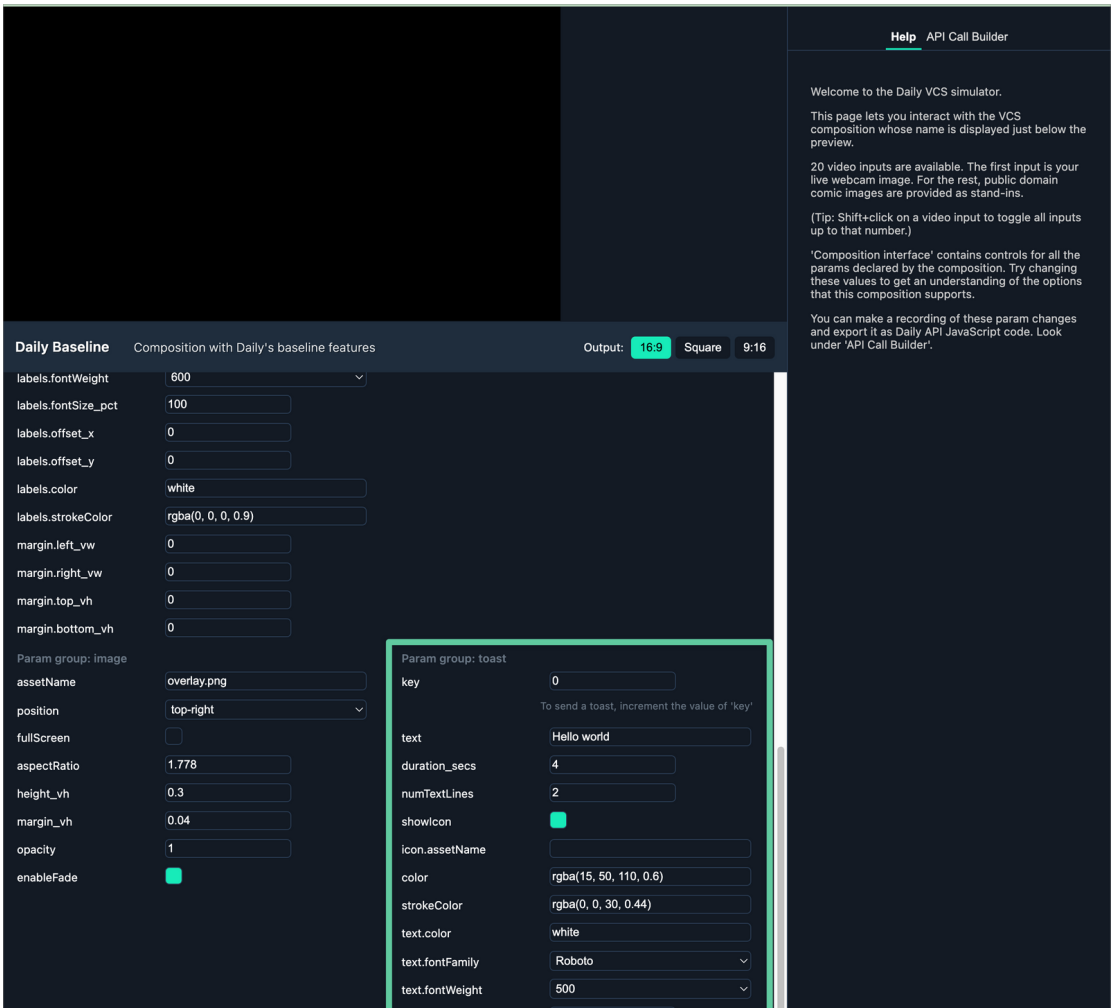
The toast component itself can be customized in several ways based on the options you set in the `updateLiveStreaming()` method.

For example, you can set the:

- background color
- text color
- text styles, like font weight and font family
- whether an icon should be included or not and, if so, which icon to use
- how long the toast should be displayed

And more! To see a full list of toast options, read our [baseline composition docs].

To test these options live, try them out in the [VCS Simulator].





VCS Simulator with the toast component options highlighted

Wrapping up

To learn more about VCS, read our guide for more information. We have some exciting announcements coming up soon related to VCS, so keep an eye on our Twitter!

Get updates from Daily

Don’t miss a thing! Subscribe to our monthly newsletter and we’ll keep you up-to-date with the latest product releases, tutorials, and more from Daily.

Email*

Country*

Join peerConnection - our video/WebRTC developer community.

By clicking submit below, you consent to allow Daily.co to store and process the personal information submitted above to provide you the content requested.

Submit

More in VCS

Add a frame to your live stream host's video with Daily's VCS

8 Aug 2022 - 5 min read

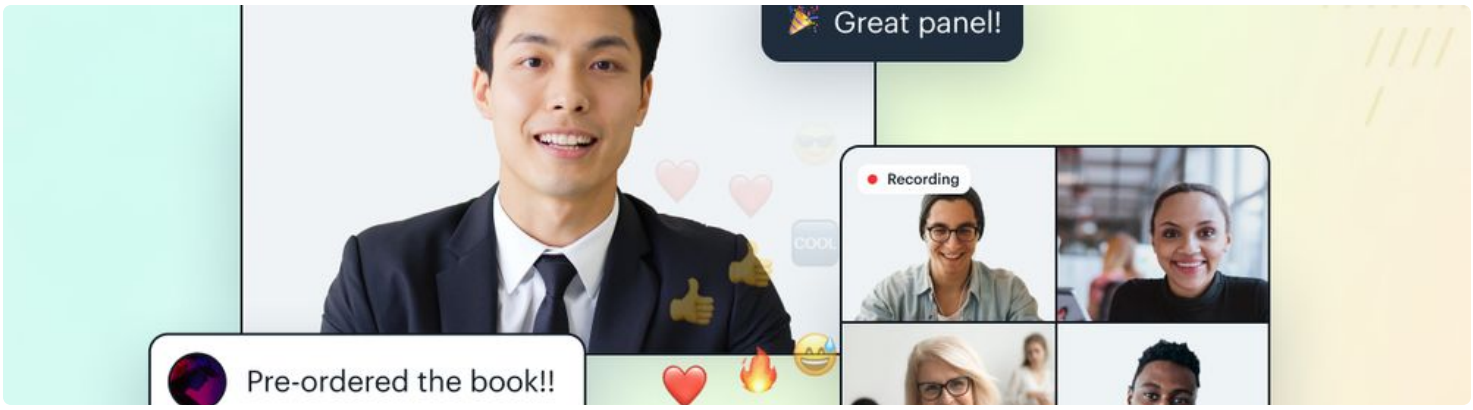
Add a splash screen to live streams and cloud recordings with Daily's VCS baseline composition

14 Jul 2022 - 7 min read

VCS tutorial: Add a dynamic moving watermark to any live stream or recording

7 Jul 2022 - 17 min read

See all 5 posts →



VCS

New beta: VCS — Build dynamic graphics and layouts for live streams and recordings

VCS technical lead Pauli Olavi Ojala discusses how Daily's Video Component System (VCS) lets developers build animated graphics, custom layouts, and event-driven components into multi-party live streams and recorded content.

Pauli Olavi Ojala, Daily Team
7 Jul 2022 – 8 min read



Daily React Hooks

Add chat to your custom video app with Daily React Hooks (Part 3)

Use daily-js and the Daily React Hooks library to add chat to your WebRTC video application



Nienke Dekker
27 Jun 2022 – 11 min read