



v 2.5.2

Una biblioteca de consejos de herramientas JS vainilla altamente personalizable y biblioteca de popover



Desarrollado por Popper.js



Star

4,395

[v1 documentos](#)

Defecto

La sugerencia de sugerencia tippy por defecto se ve así cuando no se le dan opciones.
¡Tiene una animada animación de fondo ingeniosa!

[¡Pruébame!](#)



Colocación

Una información sobre herramientas se puede colocar de cuatro maneras diferentes en relación con su elemento de referencia. Además, la información sobre herramientas se cambiará.

Parte superior

Fondo

Izquierda

Derecha

Top-Start

Extremo superior



Flechas

Las flechas apuntan hacia el elemento de referencia. Hay dos tipos diferentes de flechas: aguda y redonda. Puedes transformar la proporción y la escala de las flechas como quieras.

Defecto

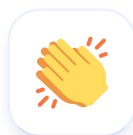
Redondo

Amplio

Flaco

Pequeña

Grande



Disparadores

Los desencadenantes definen los tipos de eventos que provocan que se muestre una información sobre herramientas.

Hover o tocar

Enfoque o toque

Hacer clic

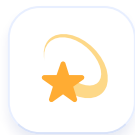


Interactividad

La información sobre herramientas puede ser interactiva, lo que significa que no se ocultará cuando coloque el cursor sobre ellas o haga clic en ellas.

Interactivo (hover)

Interactivo (click)



Animaciones

La información sobre herramientas puede tener diferentes tipos de animaciones.

Cambiar de lugar

Cambiar hacia

Descolorarse

Escala

Perspectiva

Inercia (alejamiento)

Inercia (cambio hacia adelante)

Inercia (escala)

Inercia (perspectiva)



Duración

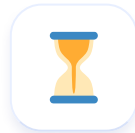
Un tippy puede tener diferentes duraciones de transición.

0ms

200ms

1000 ms

[500ms, 200ms]



Retrasar

La información sobre herramientas puede retrasar mostrar u ocultar * después de un desencadenador.

100 ms

500ms

[500ms, 200ms]



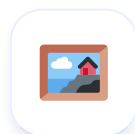
Delegación de eventos v2.1

Vincula una instancia Tippy a un contenedor primario y agrega libremente nuevos elementos secundarios sin necesidad de crear instancias Tippy para ellos.

Título compartido

Título Personalizado

Opciones personalizadas



HTML

La información sobre herramientas puede contener HTML, lo que le permite crear impresionantes popovers interactivos.

Plantillas HTML



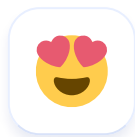
Temas

¡Un tippy puede tener cualquier tipo de tema que desee! Crear un tema personalizado es muy sencillo.

Translúcido

Ligero

Gradiente



Misc

Tippy tiene un montón de características, y está mejorando constantemente.

Seguir el cursor

Toque y mantenga

Distancia

Compensar

Pequeña

Grande



Opción 1

Incluya este script desde el [CDN unpkg](https://unpkg.com/tippy.js@2.5.2/dist/tippy.all.min.js) en su documento HTML antes de sus propios scripts:

HTML

```
<script src="https://unpkg.com/tippy.js@2.5.2/dist/tippy.all.min.js"></script>
```

Una vez que esté cargado, tendrás acceso al `tippy` módulo que te permitirá crear increíbles consejos sobre herramientas.

opcion 2

Instalar usando npm o hilo:

SHELL

```
npm install tippy.js
```

SHELL

```
yarn add tippy.js
```

Entonces puedes importar el `tippy` módulo:

JAVASCRIPT

```
// Node environment
const tippy = require('tippy.js')

// With a module bundler (webpack/rollup/parcel)
import tippy from 'tippy.js'
```

Archivos

Tippy crea varios archivos diferentes que se pueden usar:

- `tippy.all.js` es todas las dependencias (Tippy + Popper + CSS) en un solo archivo. El CSS se inyecta en el encabezado del documento.

- `tippy.js` Tippy + Popper juntos, sin el CSS.
- `tippy.standalone.js` es Tippy en sí mismo, sin Popper o CSS.
- `tippy.css` es la hoja de estilo CSS de Tippy en sí misma.

También hay `.min` versiones de lo anterior, lo que significa que el archivo está minimizado para uso de producción.



1. Agregue el contenido de su información sobre herramientas 🛠️

Primero, déle a su (s) elemento (s) de referencia un atributo de título que contenga su contenido de información sobre herramientas.

HTML

```
<button class="btn" title="I'm a tooltip!">Text</button>
```

Si pasa el cursor sobre el botón, notará que la información sobre herramientas predeterminada del navegador (generalmente la información sobre herramientas del sistema operativo nativo) aparece después de un retraso.

2. Crea un tippy 🌟

Para dar una propina a los elementos, deberá agregar algunas etiquetas JavaScript dentro de su página HTML justo antes de la etiqueta de cierre . `script` `body`

HTML

```
<script>
tippy('.btn')
</script>
```

Resultado:

Texto

Cuando `tippy()` se invoca la función y se le da una cadena de selector de CSS, encontrará todos los elementos que coinciden, comprueba si tienen un atributo `no vacío` y luego aplica su magia para darles una buena información sobre herramientas.

`title`

Mutaciones

El elemento (s) de referencia es modificado por Tippy de la siguiente manera:

HTML

```
<!-- Before -->
<button class="btn" title="I'm a tooltip!">Text</button>
<!-- After -->
<button class="btn" data-tippy data-original-title="I'm a tooltip!">Text</button>
```

- `title` atributo es eliminado
- `data-tippy` atributo se agrega
- `data-original-title` atributo se agrega que contiene la cadena `title`

Además, una vez que la información sobre herramientas ha cambiado por completo, se agrega un atributo para ally . `aria-describedby`

Tipos de entrada adicionales

Un DOM único (o una matriz de ellos) funcionará: `Element`

JAVASCRIPT

```
tippy(document.querySelector('.btn'))
```

Además de a `NodeList` :

JAVASCRIPT

```
tippy(document.querySelectorAll('.btn'))
```


v2.5 Use `tippy.one()` si está creando una sola información sobre herramientas. Esto devolverá la instancia de información sobre herramientas directamente, en lugar de un objeto de colección (porque `tippy()` puede crear varias instancias de información sobre herramientas a la vez).

JAVASCRIPT

```
tippy.one(document.querySelector('.btn'))
```

Tippify todos los elementos titulados 🔍

Use este selector :

JAVASCRIPT

```
tippy('[title]')
```

Avanzado 🧐

Puede usar un elemento virtual como referencia de posicionamiento en lugar de un elemento real:

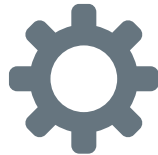
JAVASCRIPT

```
const virtualReference = {
  attributes: {
    title: "I'm a tooltip!"
  },
  getBoundingClientRect() {
    return {
      width: 100,
      height: 100,
      top: 100px,
      left: 100px,
      right: 200px,
      bottom: 200px
    }
  },
  clientHeight: 100,
  clientWidth: 100
}
```

```
}
```

```
tippy(virtualReference)
```

Popper.js usa estas propiedades para determinar la posición de la información sobre herramientas.



`tippy()` toma un objeto de opciones como segundo argumento para que personalices los tooltips que se están creando. Aquí hay un ejemplo:

JAVASCRIPT

```
tippy('.btn', {  
  delay: 100,  
  arrow: true,  
  arrowType: 'round',  
  size: 'large',  
  duration: 500  
  animation: 'scale'  
})
```

Resultado:

Texto

Atributos de datos

También puede especificar opciones en el elemento de referencia en sí agregando atributos. Esto anulará las opciones especificadas en la instancia. `data-tippy-*`

HTML

```
<button
  class="btn"
  title="I'm a Tippy tooltip!"
  data-tippy-delay="0"
  data-tippy-arrow="false"
  data-tippy-size="small"
  data-tippy-animation="shift-toward"
>
  Text
</button>
```

Resultado:

Anulado



A continuación hay una lista de todas las opciones posibles que puede proporcionar `tippy()`. Los valores son los predeterminados utilizados, con las diferentes entradas que se enumeran como un comentario al lado.

JAVASCRIPT

```
tippy(ref, {
  // Available v2.3+ - If true, HTML can be injected in the title attribute
  allowTitleHTML: true,

  // If true, the tooltip's background fill will be animated (material effect)
  animateFill: true,
```

```
// The type of animation to use
animation: 'shift-away', // 'shift-toward', 'fade', 'scale', 'perspective'

// Which element to append the tooltip to
appendTo: document.body, // Element or Function that returns an element

// Whether to display the arrow. Disables the animateFill option
arrow: false,

// Transforms the arrow element to make it larger, wider, skinnier, offset,
arrowTransform: '', // CSS syntax: 'scaleX(0.5)', 'scale(2)', 'translateX(50%)'

// The type of arrow. 'sharp' is a triangle and 'round' is an SVG shape
arrowType: 'sharp', // 'round'

// The tooltip's Popper instance is not created until it is shown for the first
// time by default to increase performance
createPopperInstanceOnInit: false,

// Delays showing/hiding a tooltip after a trigger event was fired, in ms
delay: 0, // Number or Array [show, hide] e.g. [100, 500]

// How far the tooltip is from its reference element in pixels
distance: 10,

// The transition duration
duration: [350, 300], // Number or Array [show, hide]

// If true, whenever the title attribute on the reference changes, the tooltip
// will automatically be updated
dynamicTitle: false,

// If true, the tooltip will flip (change its placement) if there is not enough
```

```
// room in the viewport to display it
flip: true,

// The behavior of flipping. Use an array of placement strings, such as
// ['right', 'bottom'] for the tooltip to flip to the bottom from the right
// if there is not enough room
flipBehavior: 'flip', // 'clockwise', 'counterclockwise', Array

// Whether to follow the user's mouse cursor or not
followCursor: false,

// Upon clicking the reference element, the tooltip will hide.
// Disable this if you are using it on an input for a focus trigger
// Use 'persistent' to prevent the tooltip from closing on body OR reference
// click
hideOnClick: true, // false, 'persistent'

// Specifies that the tooltip should have HTML content injected into it.
// A selector string indicates that a template should be cloned, whereas
// a DOM element indicates it should be directly appended to the tooltip
html: false, // 'selector', DOM Element

// Adds an inertial slingshot effect to the animation. TIP! Use a show duration
// that is twice as long as hide, such as `duration: [600, 300]`
inertia: false,

// If true, the tooltip becomes interactive and won't close when hovered over
// or clicked
interactive: false,

// Specifies the size in pixels of the invisible border around an interactive
// tooltip that prevents it from closing. Useful to prevent the tooltip
// from closing from clumsy mouse movements
interactiveBorder: 2,
```

```
// Available v2.2+ - If false, the tooltip won't update its position (or flip)
// when scrolling
livePlacement: true,

// The maximum width of the tooltip. Add units such as px or rem
// Avoid exceeding 300px due to mobile devices, or don't specify it at all
maxWidth: '',

// If true, multiple tooltips can be on the page when triggered by clicks
multiple: false,

// Offsets the tooltip popper in 2 dimensions. Similar to the distance option
// but applies to the parent popper element instead of the tooltip
offset: 0, // '50, 20' = 50px x-axis offset, 20px y-axis offset

// Callback invoked when the tooltip fully transitions out
onHidden(instance) {},

// Callback invoked when the tooltip begins to transition out
onHide(instance) {},

// Callback invoked when the tooltip begins to transition in
onShow(instance) {},

// Callback invoked when the tooltip has fully transitioned in
onShown(instance) {},

// If true, data-tippy-* attributes will be disabled for increased performance
performance: false,

// The placement of the tooltip in relation to its reference
placement: 'top', // 'bottom', 'left', 'right', 'top-start', 'top-end', etc
```

```
// Popper.js options. Allows more control over tooltip positioning and behavior
popperOptions: {},

// The size of the tooltip
size: 'regular', // 'small', 'large'

// If true, the tooltip's position will be updated on each animation frame
// the tooltip will stick to its reference element if it moves
sticky: false,

// Available v2.1+ - CSS selector string used for event delegation
target: null, // e.g. '.className'

// The theme, which is applied to the tooltip element as a class name, i.e.
// 'dark-theme'. Add multiple themes by separating each by a space, such as
// 'dark custom'
theme: 'dark',

// Changes trigger behavior on touch devices. It will change it from a tap
// to show and a tap off to hide, to a touch-and-hold to show, and a release
// to hide
touchHold: false,

// The events on the reference element which cause the tooltip to show
trigger: 'mouseenter focus', // 'click', 'manual'

// Transition duration applied to the Popper element to transition between
// position updates
updateDuration: 350,

// The z-index of the popper
zIndex: 9999
})
```

Modificando las opciones predeterminadas

Puede modificar las opciones accediendo a ellas a través de `tippy.defaults`, que se aplicará a cada instancia futura.

Más control sobre la información sobre herramientas

Especifique una `popperOptions` propiedad con las opciones de Popper.js. Vea la [documentación de Popper.js](#) para ver todas las opciones que puede especificar.

Devolución de llamada

Si desea que las cosas sucedan durante la demostración y ocultación de eventos de información sobre herramientas, puede especificar funciones de devolución de llamada en el objeto de opciones.

JAVASCRIPT

```
tippy(ref, {
  onShow(instance) {
    // When the tooltip begins to transition in
  },
  onShown(instance) {
    // When the tooltip has fully transitioned in
  },
  onHide(instance) {
    // When the tooltip begins to transition out
  },
  onHidden(instance) {
    // When the tooltip has fully transitioned out and is removed from the DOM
  },
  wait(show, event) {
    // Delays showing the tooltip until you manually invoke show()
  }
})
```


Información sobre herramientas AJAX

Las llamadas le permiten hacer cosas poderosas con información sobre herramientas. Aquí hay un ejemplo de contenido dinámico que, en el show, obtiene una nueva imagen aleatoria de Unsplash API. Nota: esto requiere un navegador que admita la nueva API de búsqueda.

Desplácese por una nueva imagen

[Demo de CodePen](#)

Delegación de eventos v2.1

La delegación de eventos solo requiere una configuración mínima. Su configuración debe ser similar a esto, con un elemento principal que envuelva los elementos secundarios a los que le gustaría dar información sobre herramientas:

HTML


```
<div id="parent" title="Shared title">
  <div class="child">Text</div>
  <div class="child">Text</div>
  <div class="child">Text</div>
  <div class="other">Text</div>
</div>
```

Luego, especifique un selector de CSS como el `target` que coincide con elementos secundarios que deberían recibir información sobre herramientas

JAVASCRIPT

```
tippy('#parent', {
  target: '.child'
})
```

Nota

 Evite vincular una instancia de Tippy al cuerpo, ya que los `mouseover` / `mouseout` eventos se dispararán constantemente a medida que el cursor se mueve sobre la página. En cambio, dele al elemento padre posible más cercano.

Destruyendo una instancia de delegado

Cuando destruyes la instancia Tippy de un delegado, también destruirá las instancias Tippy de todos los niños objetivo. Para deshabilitar este comportamiento, pase `false` al método. `destroy()`

JAVASCRIPT

```
const parent = document.querySelector('#parent')
tippy(parent, { target: '.child' })
// Will not destroy any child target instances (if they had been created)
parent._tippy.destroy(false)
```

Si se especifica la opción de destino, las referencias principales se convierten en delegadas y reciben un `data-tippy-delegate` atributo en lugar de `data-tippy`

HTML

```
<div id="parent" title="Shared title" data-tippy-delegate></div>
```

Información sobre herramientas dentro de un contenedor desplazable

Agregue las siguientes opciones para que la información sobre herramientas no permanezca bloqueada dentro de la ventana gráfica.

JAVASCRIPT

```
tippy('.mySelector', {
  appendTo: document.querySelector('.mySelector').parentNode,
  popperOptions: {
    modifiers: {
      preventOverflow: {
        enabled: false
      },
      hide: {
        enabled: false
      }
    }
  }
})
```

Deshabilitar información sobre herramientas en dispositivos táctiles

Puede ser complicado determinar los dispositivos táctiles con precisión, especialmente si se tiene en cuenta la existencia de dispositivos híbridos (una combinación de mouse y entrada táctil). Simplemente detectar el agente de usuario no es suficiente.

Un usuario puede alternar entre cualquier tipo de entrada en cualquier momento, por lo que la detección de entrada dinámica está habilitada. Puede conectar la detección de Tippy de los cambios en la entrada del usuario definiendo la siguiente función de devolución de llamada:

JAVASCRIPT

```
tippy.browser.onUserInputChange = type => {  
  console.log('The user is now using', type, 'as an input method')  
}
```

Siempre que el usuario cambie su método de entrada, puede reaccionar dentro de la función de devolución de llamada. Para desactivar la información sobre herramientas para la entrada táctil pero mantenerlas habilitadas para la entrada del mouse, puede hacer lo siguiente:

JAVASCRIPT

```
const tip = tippy('[title]')  
  
tippy.browser.onUserInputChange = type => {  
  const method = type === 'touch' ? 'disable' : 'enable'  
  for (const tooltip of tip.tooltips) {  
    tooltip[method]()  
  }  
}
```

Ocultar información sobre herramientas en desplazamiento

Debido a la forma en que los navegadores disparan los `mouseleave` eventos, puede ser conveniente ocultar la información sobre herramientas y desactivar inmediatamente los detectores de eventos siempre que se produzca el

desplazamiento. Esto también podría ayudar a reducir la intromisión de una información sobre herramientas en dispositivos táctiles de pantalla pequeña, ya que comenzará a ocultarse cuando se desplacen, en lugar de cada vez que toquen en otro lugar.

JAVASCRIPT

```
window.addEventListener('scroll', () => {  
  for (const popper of document.querySelectorAll('.tippy-popper')) {  
    const instance = popper._tippy  
  
    if (instance.state.visible) {  
      instance.popperInstance.disableEventListeners()  
      instance.hide()  
    }  
  }  
})
```

Obtenga todas las instancias de Tippy

Obtener todas las instancias Tippy (no destruidas) en el documento se puede hacer en una sola línea:

JAVASCRIPT

```
Array.from(document.querySelectorAll('[data-tippy]'), el => el._tippy)
```

Esto devuelve una matriz que contiene todas las instancias actuales de Tippy (excluyendo delegados). Para incluir delegados, use este selector:

JAVASCRIPT

```
'[data-tippy], [data-tippy-delegate]'
```



`Array.from` necesita un polyfill para navegadores más antiguos.



Es importante distinguir entre el objeto devuelto de una llamada y una instancia de Tippy. Cuando llamas , puede crear múltiples información sobre herramientas (instancias de Tippy) a la vez. `tippy()` `tippy()`

Las instancias Tippy se refieren a información sobre herramientas individuales, mientras que el objeto devuelto hace referencia a la colección. `tippy()`

`tippy()` objeto

JAVASCRIPT

```
const tip = tippy('.btn')
```

`tip` es un objeto simple

JAVASCRIPT

```
{
  // selector that was supplied to tippy()
  selector: '.btn',

  // default + instance options merged together
  options: { ... },

  // Array of all Tippy instances that were created
  tooltips: [Tippy, Tippy, Tippy, ...],

  // Method to destroy all the tooltips that were created
  destroyAll() { }
}
```

Instancias Tippy

Almacenado en elementos de referencia a través de la `_tippy` propiedad y dentro de la matriz del objeto. `tooltips` `tippy()`

```
tippy('.btn')  
const btn = document.querySelector('.btn')  
const tipInstance = btn._tippy
```

`tipInstance` es una instancia Tippy.

```
{  
  // id of the Tippy instance (1 to Infinity)  
  id: 1,  
  
  // Popper element that contains the tooltip  
  popper: Element,  
  
  // Popper instance is not created until shown for the first time,  
  // unless specified otherwise  
  popperInstance: null,  
  
  // Reference element that is the trigger for the tooltip  
  reference: Element,  
  
  // Array of objects containing the event + handler function of each trigger  
  listeners: [{ ... }, { ... }, ...],  
  
  // Defaults + instance + attribute options merged together  
  options: { ... },  
  
  // The state of the tooltip  
  state: {  
    // Has the instance been destroyed?  
    destroyed: false,  
    // Is the instance enabled?  
    enabled: true,  
    // Is the tooltip currently visible and not transitioning out?  
    visible: false
```

```
},  
  
// title content of the tooltip (null if HTML)  
title: 'example'  
}
```

Atajos

Hay varios atajos disponibles para acceder a la instancia.

JAVASCRIPT

```
// The popper element has the instance attached to it:  
popper._tippy  
  
// As does the reference element (as seen above):  
reference._tippy  
  
// The popper also has the reference directly attached:  
popper._reference
```



Las instancias de Tippy tienen 5 métodos disponibles que le permiten controlar la información sobre herramientas sin el uso de eventos de IU. Son:

- `Tippy.prototype.show()`
- `Tippy.prototype.hide()`
- `Tippy.prototype.enable()`
- `Tippy.prototype.disable()`
- `Tippy.prototype.destroy()`

Dado el siguiente elemento con una información sobre herramientas:

HTML

```
<button title="Hello!">Text</button>
```

JAVASCRIPT

```
const btn = document.querySelector('button')  
tippy(btn)
```

La instancia de Tippy se almacena en el elemento de botón a través de la `_tippy` propiedad.

v2.5 Si está tratando con un solo elemento / información sobre herramientas, puede usar el `tippy.one()` método para devolver directamente la instancia en lugar de tener que usar la `_tippy` propiedad.

JAVASCRIPT

```
const instance = tippy.one('button')
```

Mostrar la información sobre herramientas

JAVASCRIPT

```
btn._tippy.show()
```

Ocultar la información sobre herramientas

JAVASCRIPT

```
btn._tippy.hide()
```

Duración de transición personalizada

Pase un número como argumento para anular la opción de instancia:

JAVASCRIPT

```
btn._tippy.show(200) // 200ms  
btn._tippy.hide(1000) // 1000ms
```

Deshabilitar la información sobre herramientas

La información sobre herramientas puede desactivarse temporalmente para mostrar / ocultar:

JAVASCRIPT

```
btn._tippy.disable()
```

Para volver a habilitar:

JAVASCRIPT

```
btn._tippy.enable()
```

Destruye la información sobre herramientas

Para destruir permanentemente la información sobre herramientas y eliminar todos los oyentes del elemento de referencia:

JAVASCRIPT

```
btn._tippy.destroy()
```

La `_tippy` propiedad se elimina del elemento de referencia después de la destrucción.

Actualiza la información sobre herramientas

No hay ningún método para actualizar el contenido de la información sobre herramientas, ¡porque es fácil de hacer!

Opción 1 (recomendado): cambie el título en el elemento de referencia y use la opción:

`dynamicTitle`

JAVASCRIPT

```
tippy(btn, { dynamicTitle: true })  
btn.title = 'New tooltip :)'
```

Opción 2: actualizar manualmente la información sobre herramientas:

JAVASCRIPT

```
btn._tippy.popper.querySelector('.tippy-content').textContent = 'New tooltip'
```

Nota: con el método anterior, `_tippy.title` no reflejará el nuevo título, deberá actualizarlo manualmente.

Si está utilizando una plantilla HTML, guárdela en una referencia de variable para modificarla más adelante.

```
const template = document.querySelector('template')
tippy(btn, { html: template })
template.textContent = 'New tooltip :)'
```



Hay dos formas de crear una plantilla HTML: **clonación** o **referencia directa** .

Opción 1: Clonación

Clona la plantilla, `innerHTML` pero no la modifica.

Opción: `html: '#templateId'` selector que coincide con una plantilla en el documento

- Reutilizable
- Permanece en la página
- No guarda los oyentes del evento conectados a él
- No directamente modificable

Opción 2: referencia directa

Anexa directamente un elemento a la información sobre herramientas.

Opción: `html: document.querySelector('#templateId')` HTMLElement

- Solo se puede usar una vez
- Eliminado de la página y anexo al elemento de información sobre herramientas
- Guarda oyentes del evento conectados a él
- Directamente modificable

En el documento o en JavaScript en alguna parte, crea una plantilla.

Clonación

HTML

```
<div id="myTemplate" style="display: none;">
  <h3>Cool <span style="color: pink;">HTML</span> inside here!</h3>
</div>
```

Referencia directa del elemento

HTML

```
<div id="myTemplate">
  <h3>Cool <span style="color: pink;">HTML</span> inside here!</h3>
</div>
```

Elemento dinámico con JS

JAVASCRIPT

```
const myTemplate = document.createElement('div')
myTemplate.innerHTML = '<h3>Cool <span style="color: pink;">HTML</span> inside here!</h3>'
```

Luego, especifique una `html` opción, eligiendo una de las opciones.

JAVASCRIPT

```
tippy('selector', {
  html: '#myTemplate',
  // ...or...
  html: document.querySelector('#myTemplate'),
  // ...or you can clone a direct element too...
  html: document.querySelector('#myTemplate').cloneNode(true)
})
```

Resultado:

¡Tengo una plantilla HTML!



¡Crear un tema para su información sobre herramientas es fácil! Si quería hacer un tema llamado `honeybee`, entonces su CSS se vería así:

```
.tippy-tooltip.honeybee-theme {  
  /* Your styling here. Example: */  
  background-color: yellow;  
  border: 2px solid orange;  
  font-weight: bold;  
  color: #333;  
}
```

CSS

El `-theme` sufijo es obligatorio.

Diseñando el `animateFill` telón de fondo

De manera predeterminada, los consejos de ayuda tippy tienen una animación genial para llenar el fondo, que es solo un círculo que se expande. Su nombre de clase es `tippy-backdrop`:

```
.tippy-tooltip.honeybee-theme .tippy-backdrop {  
  /* Your styling here. Example: */  
  background-color: yellow;  
}
```

CSS

Si está utilizando la animación de fondo, evite diseñar la información sobre herramientas directamente, solo en el fondo.

Diseñando la flecha

Hay dos selectores de flecha: `.tippy-arrow` y `.`. El primero es la forma de triángulo puro de CSS, mientras que el segundo es un SVG personalizado. `.tippy-roundarrow`

CSS

```
.tippy-popper[x-placement^=top] .tippy-tooltip.honeybee-theme .tippy-arrow {  
  /* Your styling here. */  
}
```

Tendrá que aplicarle un estilo a la flecha para cada ubicación de popper diferente (arriba, abajo, izquierda, derecha), por lo que el selector es tan largo.

Diseñando el contenido directamente

CSS

```
.tippy-tooltip.honeybee-theme .tippy-content {  
  /* Your styling here. Example: */  
  color: #333;  
}
```

Especifica una `theme` opción

Para ver cómo se ve tu tema genial, especifica una `theme` opción para tippy:

JAVASCRIPT

```
tippy('.btn', {  
  theme: 'honeybee',  
  // ...or add multiple themes by separating each by a space...  
  theme: 'honeybee bumblebee shadow'  
})
```

`.honeybee-theme` , `.bumblebee-theme` y son los selectores para esta lista de temas.
`.shadow-theme`

Resultado:

Tema personalizado



Soporte actual (rastreado): **96% Global , 99% EE. UU.**

Tippy admite navegadores `requestAnimationFrame` y soporte: vea [datos de canius](#) .
`MutationObserver`

IE10 solo se admite parcialmente, a menos que se realice el relleno
`MutationObserver` , entonces es totalmente compatible. `dynamicTitle` se basa en eso.



En un 2016 MacBook Pro 2.6 GHz Skylake, utilizando Chrome 65:

- **Modo de rendimiento desactivado:** 13 ms por cada 100 elementos
- **Modo de rendimiento activado:** 6 ms por cada 100 elementos
- **Delegación de evento:** <1 ms por 1 elemento!