 Descripción general de elementos angulares

*Los elementos angulares* son componentes angulares empaquetados como *elementos personalizados* (también llamados web components), un estándar web para definir nuevos elementos HTML de una manera independiente del marco.

[Los elementos personalizados](https://developer.mozilla.org/en-US/docs/Web/Web_Components/Using_custom_elements) son una característica de la plataforma web actualmente compatible con Chrome, Firefox, Opera y Safari, y disponible en otros navegadores a través de polyfills (consulte [Soporte del navegador](https://angular.io/guide/elements#browser-support) ). Un elemento personalizado extiende HTML al permitirle definir una etiqueta cuyo contenido es creado y controlado por código JavaScript. El navegador mantiene una CustomElementRegistry de elementos personalizados definidos, que asigna una clase de JavaScript instanciable a una etiqueta HTML.

El @angular/elementspaquete exporta una [createCustomElement](https://angular.io/api/elements/createCustomElement)()API que proporciona un puente desde la interfaz de componentes de Angular y la funcionalidad de detección de cambios a la API DOM incorporada.

La transformación de un componente en un elemento personalizado hace que toda la infraestructura angular requerida esté disponible para el navegador. Crear un elemento personalizado es simple y directo, y conecta automáticamente su vista definida por componentes con detección de cambios y enlace de datos, asignando la funcionalidad angular a los equivalentes HTML nativos correspondientes.

## **Elementos personalizados**

Los elementos personalizados arrancan ellos mismos: se inician automáticamente cuando se agregan al DOM y se destruyen automáticamente cuando se eliminan del DOM. Una vez que se agrega un elemento personalizado al DOM para cualquier página, se ve y se comporta como cualquier otro elemento HTML, y no requiere ningún conocimiento especial de términos angulares o convenciones de uso.

* **Contenido dinámico fácil en una aplicación angular**

La transformación de un componente en un elemento personalizado proporciona una ruta fácil para crear contenido HTML dinámico en su aplicación Angular. El contenido HTML que agrega directamente al DOM en una aplicación Angular normalmente se muestra sin procesamiento Angular, a menos que defina un componente dinámico , agregue su propio código para conectar la etiqueta HTML a los datos de su aplicación y participe en la detección de cambios. Con un elemento personalizado, todo ese cableado se realiza automáticamente.

* **Aplicaciones ricas en contenido**

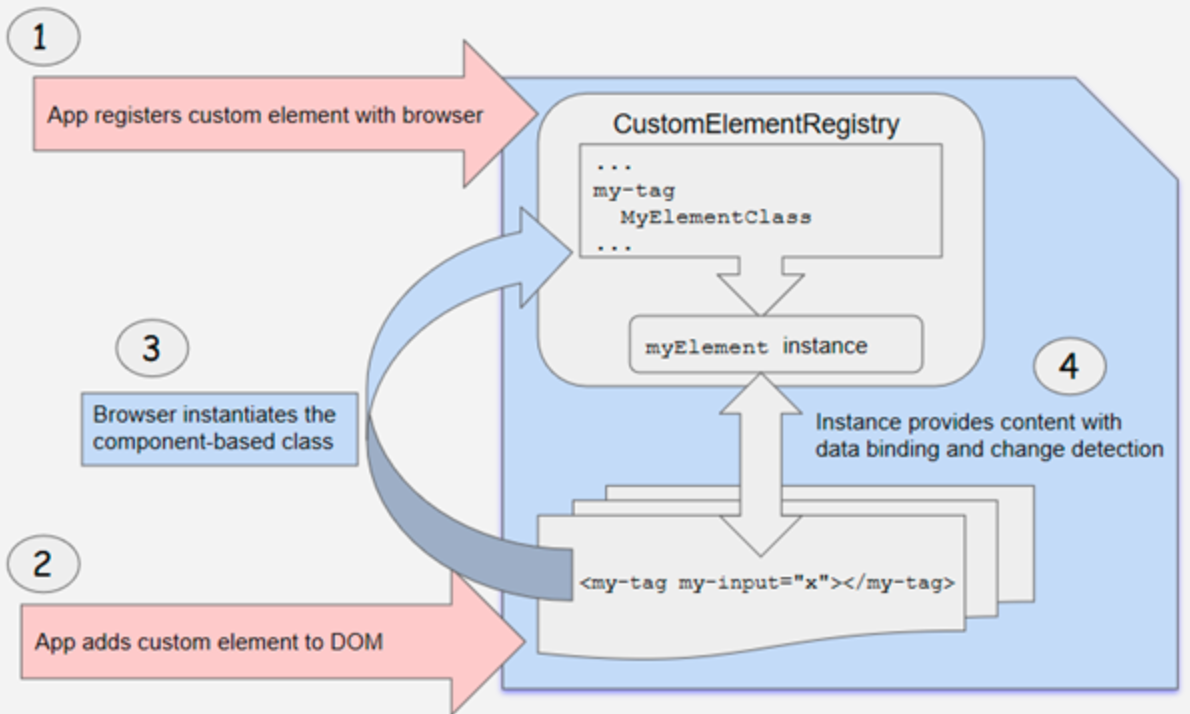
Si tiene una aplicación rica en contenido, como la aplicación Angular que presenta esta documentación, los elementos personalizados le permiten brindar a sus proveedores de contenido funcionalidades sofisticadas de Angular sin requerir conocimiento de Angular. Por ejemplo, las herramientas de navegación angular agregan una guía angular como esta directamente al DOM, pero puede incluir elementos especiales como el <code-snippet>que realiza operaciones complejas. Todo lo que necesita decirle a su proveedor de contenido es la sintaxis de su elemento personalizado. No necesitan saber nada sobre Angular, ni nada sobre las estructuras de datos o la implementación de sus componentes.

### **Cómo funciona**

Use la [createCustomElement](https://angular.io/api/elements/createCustomElement)() función para convertir un componente en una clase que se pueda registrar con el navegador como elemento personalizado. Después de registrar su clase configurada con el registro de elementos personalizados del navegador, puede usar el nuevo elemento como un elemento HTML incorporado en el contenido que agrega directamente al DOM:

   <my-button title="Botón de pago webcomponents"></my-button>

Cuando su elemento personalizado se coloca en una página, el navegador crea una instancia de la clase registrada y la agrega al DOM. El contenido lo proporciona la plantilla del componente, que utiliza la sintaxis de la plantilla angular, y se representa con el componente y los datos DOM. Las propiedades de entrada en el componente corresponden a los atributos de entrada para el elemento.

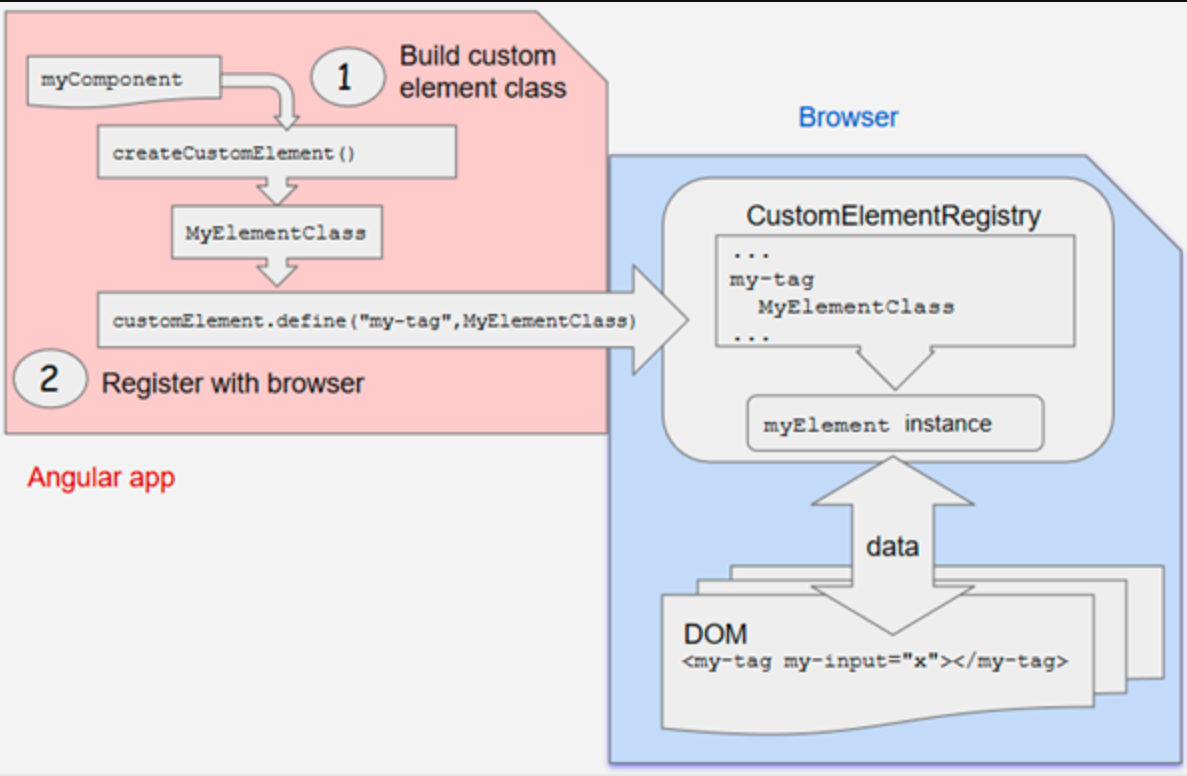


## **Transformación de componentes a elementos personalizados**

Angular proporciona la [createCustomElement](https://angular.io/api/elements/createCustomElement)() función para convertir un componente angular, junto con sus dependencias, en un elemento personalizado. La función recopila las propiedades observables del componente, junto con la funcionalidad angular que el navegador necesita para crear y destruir instancias, y para detectar y responder a los cambios.

El proceso de conversión implementa la [NgElementConstructor](https://angular.io/api/elements/NgElementConstructor) interfaz y crea una clase de constructor que está configurada para producir una instancia de arranque automático de su componente.

Use una función de JavaScript,  customElements.define()para registrar el constructor configurado y su etiqueta de elemento personalizado asociada con la del navegador CustomElementRegistry. Cuando el navegador encuentra la etiqueta para el elemento registrado, usa el constructor para crear una instancia de elemento personalizado.



### **mapeo**

Un elemento personalizado aloja un componente angular, proporcionando un puente entre los datos y la lógica definidos en el componente y las API DOM estándar. Las propiedades de los componentes y los mapas lógicos directamente en los atributos HTML y el sistema de eventos del navegador.

* La API de creación analiza el componente en busca de propiedades de entrada y define los atributos correspondientes para el elemento personalizado. Transforma los nombres de propiedad para hacerlos compatibles con elementos personalizados, que no reconocen distinciones entre mayúsculas y minúsculas. Los nombres de atributos resultantes usan minúsculas separadas por guiones. Por ejemplo, para un componente con , el elemento personalizado correspondiente define un atributo .@[Input](https://angular.io/api/core/Input)('myInputProp') inputPropmy-input-prop
* Las salidas de componentes se envían como [eventos personalizados](https://developer.mozilla.org/en-US/docs/Web/API/CustomEvent) HTML , con el nombre del evento personalizado que coincide con el nombre de salida. Por ejemplo, para un componente con , el elemento personalizado correspondiente enviará eventos con el nombre "valueChanged", y los datos emitidos se almacenarán en la propiedad del evento . Si proporciona un alias, se utiliza ese valor; por ejemplo, resulta en eventos de despacho con el nombre "myClick".@[Output](https://angular.io/api/core/Output)() valueChanged = new [EventEmitter](https://angular.io/api/core/EventEmitter)()detail@[Output](https://angular.io/api/core/Output)('myClick') clicks = new [EventEmitter](https://angular.io/api/core/EventEmitter)<string>();

Para obtener más información, consulte la documentación del componente web para [crear eventos personalizados](https://developer.mozilla.org/en-US/docs/Web/Guide/Events/Creating_and_triggering_events#Creating_custom_events) .

## **Soporte de navegador para elementos personalizados**

La característica de la Plataforma Web de [elementos personalizados](https://developer.mozilla.org/en-US/docs/Web/Web_Components/Using_custom_elements) recientemente desarrollada actualmente es compatible de forma nativa en varios navegadores. El soporte está pendiente o planificado en otros navegadores.

|  |  |
| --- | --- |
| **Navegador** | **Soporte de elementos personalizados** |
| Crome | Apoyado de forma nativa. |
| Ópera | Apoyado de forma nativa. |
| Safari | Apoyado de forma nativa. |
| Firefox | Compatible de forma nativa a partir de la versión 63. En versiones anteriores: establezca las preferencias dom.webcomponents.enabledy dom.webcomponents.customelements.enableden verdadero. |
| Edge | Trabajando en una implementación. |

En los navegadores que admiten elementos personalizados de forma nativa, la especificación requiere que los desarrolladores usen clases ES2015 para definir elementos personalizados; los desarrolladores pueden optar por esto configurando la [target](https://angular.io/api/router/RouterLinkWithHref#target): "es2015"propiedad en sus proyectos tsconfig.json. Como el soporte de Custom Element y ES2015 puede no estar disponible en todos los navegadores, los desarrolladores pueden optar por usar un polyfill para admitir navegadores más antiguos y el código ES5.

Utilice el [angular de la CLI](https://angular.io/cli) para configurar automáticamente su proyecto con la polyfill correcta: ng add @angular/elements --name=\*your\_project\_name\*.

* Para obtener más información sobre polyfills, consulte la [documentación de polyfill](https://www.webcomponents.org/polyfills) .
* Para obtener más información sobre el soporte de navegador Angular, consulte [Soporte de navegador](https://angular.io/guide/browser-support) .