



Web Components

Netzwerk | Hama DE

https://de.hama.com/produkte/zubehoer/computer-notebook/netzwerk

+++ JOBS & KARRIERE: BEWERBEN SIE SICH HIER +++

Deutsch

☰ Produkte Service Business hama® Login Merkzettel Suche

› Startseite
› Produkte
› Zubehör
› Computer & Notebook
Netzwerk

--- beliebig sortiert ---

 00053305 Hama AC600 Nano-WLAN-USB-Stick, 2.4/5 GHz 22,99 EUR UPE Anmerken	 00053302 Hama N150 Nano-WLAN-USB-Stick, 2.4 GHz 14,99 EUR UPE Anmerken	 00049218 Hama Bluetooth-USB-Adapter, Version 4.0 C2 + EDR 12,99 EUR UPE Anmerken
 00053188 Hama Bluetooth-USB-Adapter, Version 4.0 C1 + EDR 16,99 EUR UPE Anmerken	 00115478 Hama Netzkabel für PS4, 2,5 m 7,99 EUR UPE Anmerken	 00137221 Hama Netzkabel, 5 m, Schwarz 13,49 EUR UPE Anmerken



00053305

Hama AC600 Nano-WLAN-USB-Stick, 2.4/5 GHz

22,99 EUR UPE



```
▼<div class="productlist-item callout text-center" data-equalizer-watch style="height: 316px;">
  ▼<a href="/00053305-hama-ac600-nano-wlan-usb-stick-24-5-ghz">
    ▶<div class="item-image">...</div>
    <div class="item-stock">
    </div>
    <span class="itemNo">00053305</span>
    ▼<div class="item-description">
      <h3 class="title" itemprop="name" title=" Hama&nbsp;AC600 Nano-WLAN-USB-Stick, 2.4/5 GHz
      ">
        Hama&nbsp;AC600 Nano-WLAN-USB-Stick, 2.4/5 GHz
      </h3>
    ▼<div class="overview-price">
      ▼<div class="price">
        <span class="price">22,99</span>
        <span class="currency">EUR</span>
        <span class="priceType">UPE</span>
      </div>
      </div>
    </div>
    </a>
  ▼<ul class="menu horizontal addition">
    ▼<li>
      ▶<span class="add-leaflet" data-selected="false" data-url-leaflet="https://de.hama.com/order/leaflet/00053305?quantity=1">...</span>
    </li>
  </ul>
</div>
</div>
```



00053305

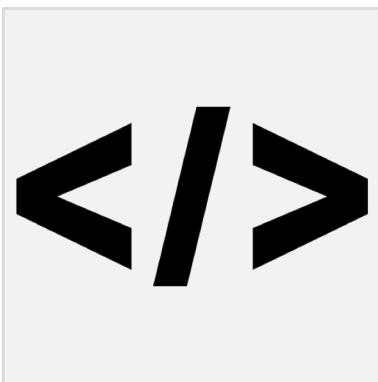
**Hama AC600 Nano-WLAN-USB-Stick,
2.4/5 GHz**

22,99 EUR UPE

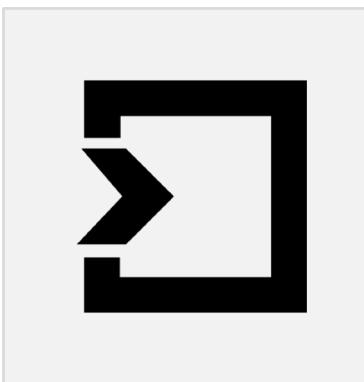


```
<Tile>
  <Item itemNo="00053305">
    <ItemPicture />
    <Icon />
    <ItemTitle />
    <ItemPrice />
    <OrderButton />
  </Item>
</Tile>
```

Web Components



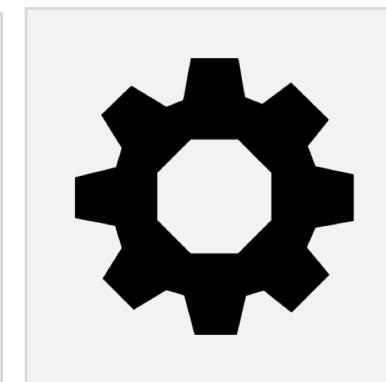
Custom
Elements



Shadow
DOM

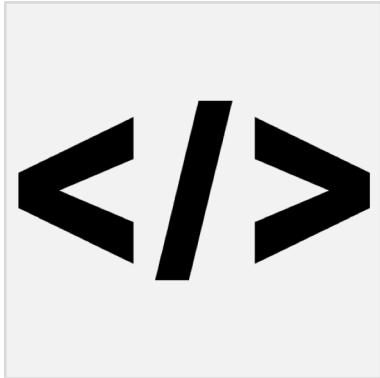


HTML
Imports

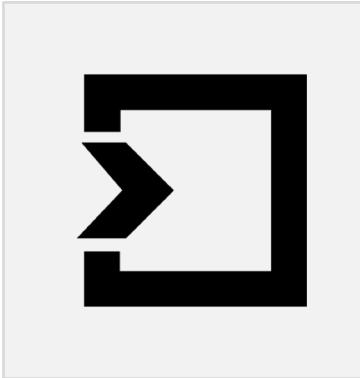


HTML
Templates

Web Components



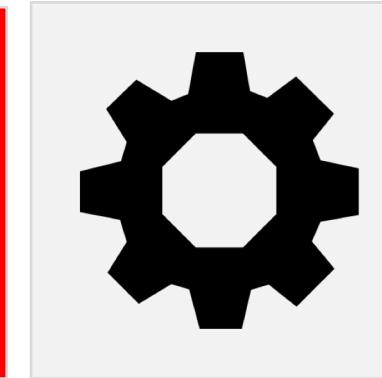
Custom
Elements



Shadow
DOM



HTML
Imports
ES
Modules



HTML
Templates

⚠ [Deprecation] HTML Imports is deprecated and will be removed in M73, around March 2019. Please use ES modules instead. See <https://www.chromestatus.com/features/5144752345317376> for more details.

HANDS-ON (2): WEB COMPONENTS



Wer echte Web Components programmieren oder verwenden will:

Programmiert gegen die APIs die jetzt schon in den meisten Browsern verfügbar sind

Support: custom-elementsv1	
	Chrome for Android 69+
	Chrome 67+
	iOS Safari (limited) 10.3+
	UC Browser for Android 11.8+
	Firefox 63+
	IE None
	Opera Mini None
	Safari (limited) 10.1+
	Opera (limited) 41+
	Samsung Internet 6.2+
	Android Browser 67+

oder nimmt fertige Web Components von beziehungsweise benutzt

- Google Polymer
- Oracle JET 4.0
- Vaadin stellt ebenfalls Web Components bereit

Wichtig



Das Programmiermodell von
Angular
React
Vue

entspricht nicht dem **Web Components** Standard!

Nicht das Framework ist entscheidend, sondern die Konzepte!

Erfolgreich kann man mit jedem der genannten Frameworks sein!

Es wichtiger folgende Konzepte zu verstehen:

- SPAs
- Components
- Data Flow
- State Management
- Data Binding
- Backend API (REST APIs)
- Routing
- Languages
- Testing

Nicht das Framework ist entscheidend, sondern die Konzepte!

Erfolgreich kann man mit jedem der genannten Frameworks sein!

Es wichtiger folgende Konzepte zu verstehen:

- SPAs
- Components
- Data Flow
- State Management
- Data Binding
- Backend API (REST APIs)
- Routing
- Languages
- Testing

Single Page Application

- ein HTML-Dokument
- der ganze Rest kommt als JavaScript
- Seite ändert sich dynamisch
- Wenig bis gar keine Seitenwechsel
- Anwendung fühlt sich nach Rich Client an

Nicht das Framework ist entscheidend, sondern die Konzepte!

Erfolgreich kann man mit jedem der genannten Frameworks sein!

Es wichtiger folgende Konzepte zu verstehen:

- SPAs
 - Components ←
 - Data Flow
 - State Management
 - Data Binding
 - Backend API (REST APIs)
 - Routing
 - Languages
 - Testing
- Eigene, sprechende HTML-Tags
 - Wiederkehrendes Markup zusammenfassen
 - Struktur
 - Wiederverwendung
 - Vergleichbar mit eigenen Widgets, Controls in Rich Clients

Nicht das Framework ist entscheidend, sondern die Konzepte!

Erfolgreich kann man mit jedem der genannten

Es wichtiger folgende Konzepte zu verstehen:

- SPAs
- Components
- Data Flow
- State Management
- Data Binding
- Backend API (REST APIs)
- Routing
- Languages
- Testing



Wie gehe ich mit Daten und Zustand in
meiner Anwendung um?

Nutze ich das Local Storage meines
Browser zum Zwischenspeichern von
Daten?

Definiere ich State-Objekte oder
Properties die meine Daten halten und
reiche sie durch meinen
Komponentenbaum ?

Nutze ich Framework-spezifische
Konzepte zur Datenhaltung wie
beispielsweise das Context API von
React?

Oder nutze ich (externe) Data
Stores/State Manager wie Vuex, Redux
oder MobX?

Nicht das Framework ist entscheidend, sondern die Konzepte!

Erfolgreich kann man mit jedem der genannten Frameworks sein!

Es wichtiger folgende Konzepte zu verstehen:

- SPAs
- Components
- Data Flow
- State Management
- Data Binding
- Backend API (REST APIs)
- Routing
- Languages
- Testing

Wie verbinde ich meine Anwendungslogik
mit meinen Komponenten, so dass Daten-
oder Zustandsänderungen auch sofort an
die Oberfläche kommen beziehungsweise
umgekehrt zurück ins Datenmodell ?

One Way versus Two Way Databinding

Nicht das Framework ist entscheidend, sondern die Konzepte!

Erfolgreich kann man mit jedem der genannten Frameworks sein!

Es wichtiger folgende Konzepte zu verstehen:

- SPAs
- Components
- Data Flow
- State Management
- Data Binding
- Backend API (REST APIs) 
- Routing
- Languages
- Testing

**Welche APIs nutze ich zur Anbindung an meine
Geschäftslogik?**

**Setze ich diese mit REST oder etwa GraphQL
um?**

**Wie dokumentiere ich meine REST APIs? Nutze
ich Swagger bzw. die Open API Spec oder
RAML (Restful API Modeling Language) zur
Dokumentation?**

**Und wie rufe die die APIs aus meiner
Anwendung heraus auf? Nutze ich hierfür die
Standard Fetch API oder externe Libraries wie
Axios?**

vuejs / vue.js / Vue



**„An approachable, versatile, performant web framework
to write maintainable and testable applications“**

Also ein offenes, leicht erlernbares, anpassungsfähiges, vielseitiges und schnelles Framework.

Wichtige Vue-Bestandteile

Modul	Beschreibung
Vue Core	Kernkomponente, implementiert Vue-Objekt, Data Binding und bringt alle Werkzeuge für die Entwicklung von Komponenten mit
Vuex	State Manager. Hält Komponenten zusammen, kontrolliert Zustand, verwaltet Daten und tauscht sie zwischen Komponenten. Vergleichbar mit Redux aber nicht dasselbe . Fortgeschrittenes Feature.
Vue Router	Wer SPAs programmiert, benötigt für den Wechsel der einzelnen Seiten/Masken/Formulare einen Router...
Vue CLI	Werkzeug (Kommandozeile oder Browser) zum einfachen Erzeugen von Vue-Projekten

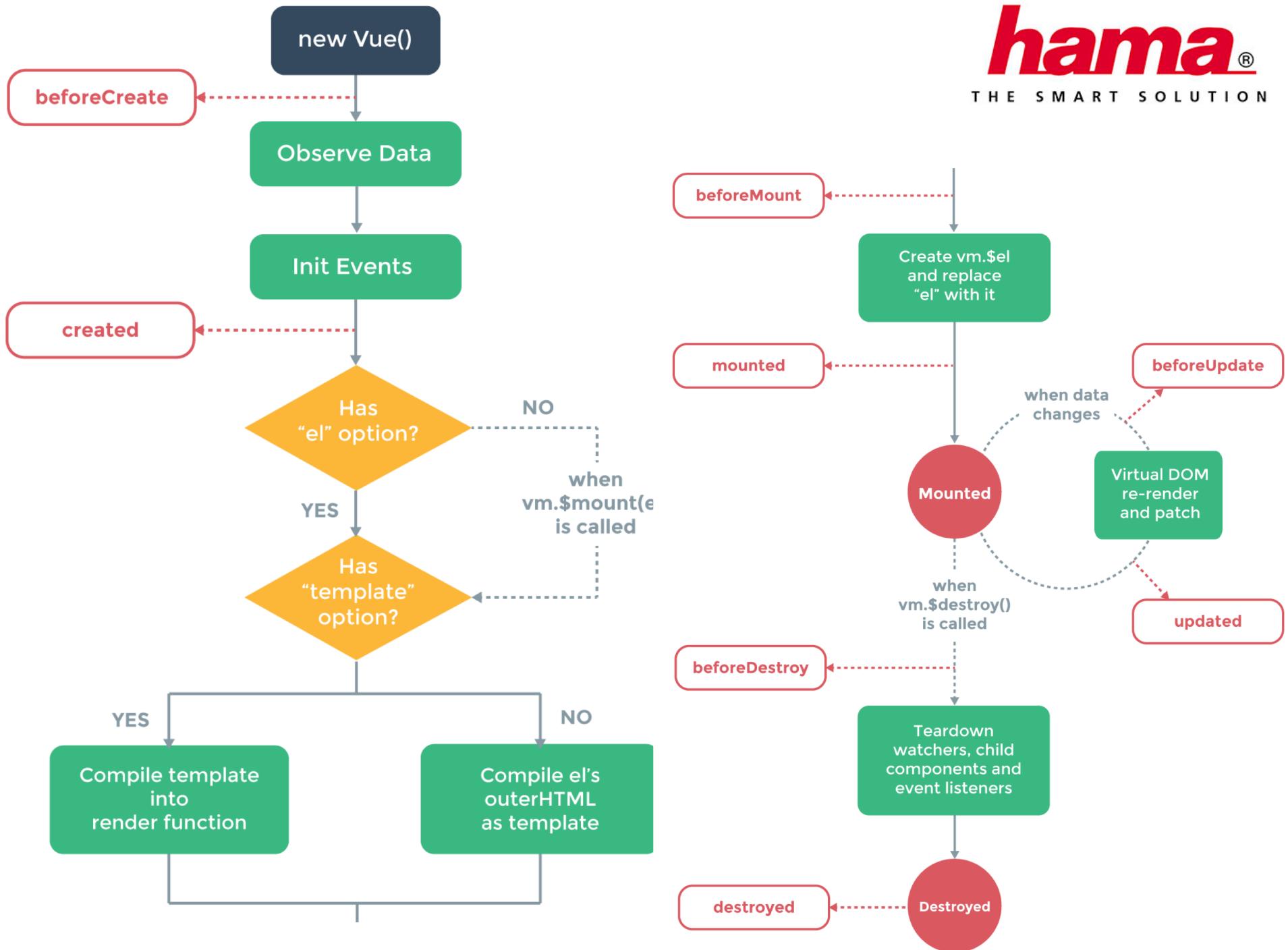
HANDS-ON (2): VUE BASICS



Vue über ein CDN einbinden

CDN-URL	Beschreibung
<code>https://cdn.jsdelivr.net/npm/vue/dist/vue.js</code>	Holt die aktuellste Vue-Version. Zur Laufzeit werden Entwickler-Informationen auf der Konsole ausgegeben.
<code>https://cdn.jsdelivr.net/npm/vue/vue.js</code>	Holt die ebenfalls die aktuellste Vue-Version. Diese Version ist jedoch minimiert und optimiert für den Einsatz in Produktion.
<code>https://cdn.jsdelivr.net/npm/vuey@2.5.15/vue/dist/vue.js</code>	Holt die Entwickler-Variante und zwar in genau der angegebenen Version. Lässt man <i>/dist</i> weg, dann wird der Produktions-Build anstelle des Entwickler-Build verwendet.

Tabelle 1: Verschiedene Varianten Vue über CDN zu beziehen



Warum Komponenten?

hama[®]
THE SMART SOLUTION



Wiederverwendung? Ja! Aber...

Das Komponentenkonzept zielt nur zum Teil darauf ab, einfach pflegbare und wiederverwendbare Einheiten von Code herzustellen.

Es geht darum, eine fachliche Anforderung, z.B. eine Anwendung zum Erfassen von Kleinanzeigen in kleine Bestandteile aufzuteilen.

Ziel ist es, die Komponentenbausteine besser überblicken zu können.

Die Anwendung wird in einer hierarchische Struktur von Software-Bausteinen aufgeteilt, aus denen sie sich Stück für Stück zusammensetzt.

Wiederverwendung nicht um jeden Preis!

- Wiederverwendung ist sinnvoll, keine Frage!
- Der Anspruch des Entwickler DARF ES NICHT SEIN Wiederverwendung und generische Routinen um jeden Preis zu entwickeln!
- Hier lauert die Gefahr, dass man sich selbst Probleme schafft und unbewusst Komplexität ins Projekt holt!
- Ganz ehrlich: Es ist kein Beinbruch, wenn Code an der einen oder anderen Stelle mal redundant gehalten wird!
- **Redundanz ist nicht per se schlecht und wenn sie einem besseren Verständnis der Software-Architektur dient ist sie mit Sicherheit auch kein Fehler!!!!**

HANDS-ON (3): VUE COMPONENTS



```

class FlagIcon extends HTMLElement {
    constructor() {
        super();
        this._shadowRoot = this.attachShadow({
            mode: 'open'
        });
        this._shadowRoot.innerHTML = `
            <link href="assets/flags.css" rel="stylesheet" type="text/css">
            `;
    }
    static get observedAttributes() {
        return ['country'];
    }
    connectedCallback() {
        this.updateFlag( this.getAttribute("country") );
    }
    attributeChangedCallback(name, oldValue, newValue) {
        this.updateFlag(newValue);
    }
    updateFlag(country) {
        this._img = this._shadowRoot.querySelector("#flag-image");
        this._img.className = 'flag flag-' + country;
    }
}
customElements.define('flag-icon', FlagIcon);

Vue.component('vue-flag-icon', {
    template: `<div>
        <link href="assets/flags.css" rel="stylesheet" type="text/css">
        
    </div>`,
    props: ['country'],
    computed: {
        css: function() {
            return 'flag flag-' + this.country;
        }
    }
});

```



HANDS-ON (4): DIVERSE ECMASCRIPT-VERSIONEN NUTZEN ERSTE SCHRITTE MIT BABELJS

