Meetup

Web Components

Einblicke in die Theorie und Praxis

Alessa Radkohl, Jan-Niklas Voß, Pascal Fecht

Online, 27.05.2020

Alessa Radkohl

Web Developer

2020

Jan-Niklas Voß

Werkstudent (Web Developer)

2018

Pascal Fecht

Software Developer

2016

Vision

Wir sehen Zukunft in Web Components, die überall lauffähig sind und über die gängigsten Frameworks hinweg funktionieren.

Hierfür entwickeln wir die *inovex elements* als open-source UI-Komponentenbibliothek.

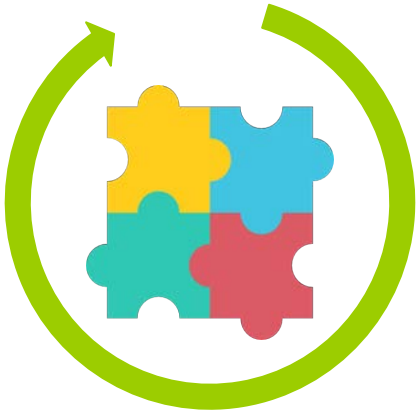
Agenda

- Web Components
  - Standards und Browser Support
  - Anwendungsszenarien
- inovex elements
  - Aufbau eines inovex elements
  - Integration in Web Frameworks
  - UI-Showcase
- Lessons Learned & Next Steps

Web Components

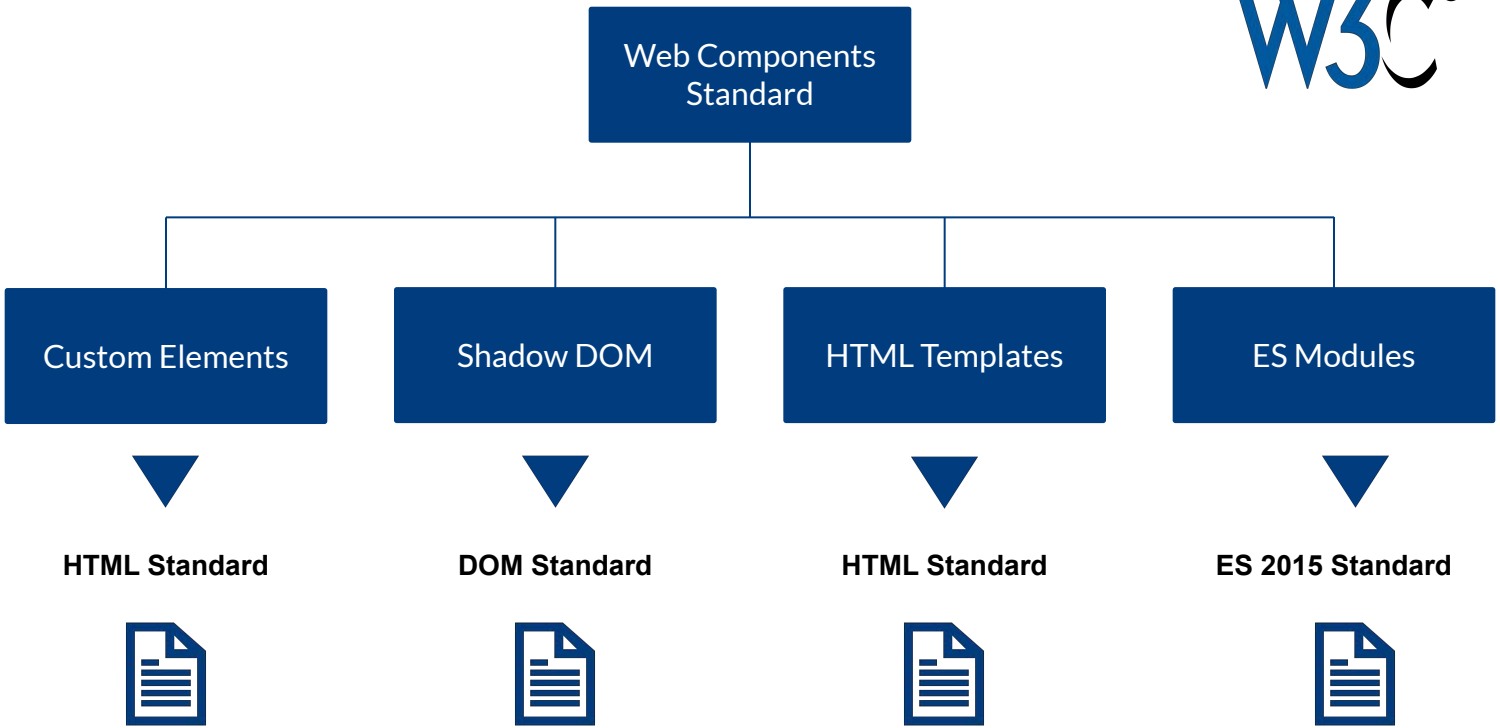
Komponenten-basierte Frontend-Entwicklung

Client-seitige JavaScript Frameworks



Web Components

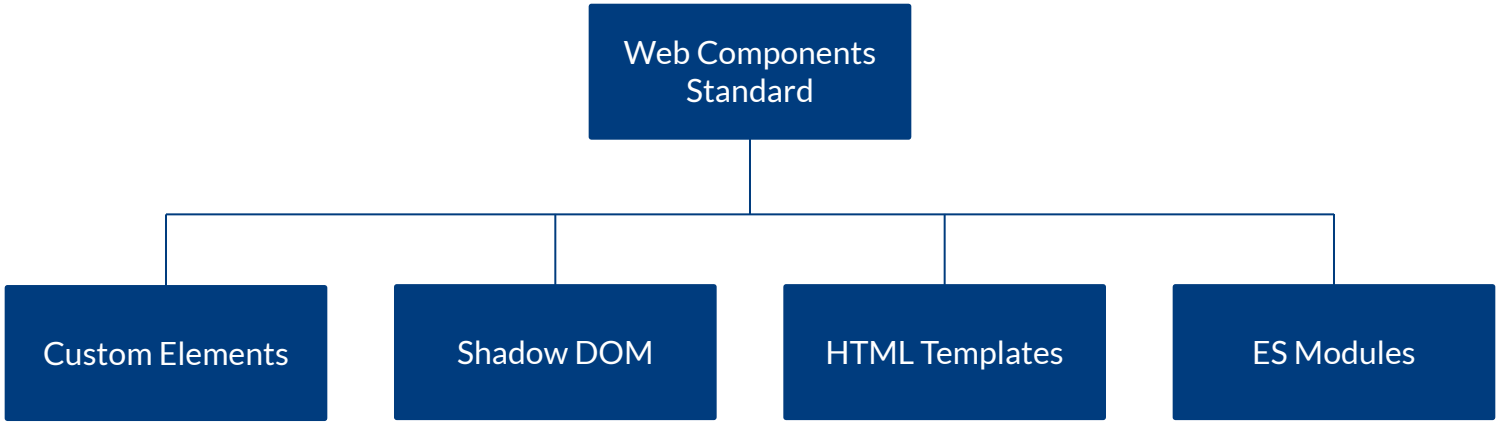
Web Components Standard



W3C®

WHATWG

Web Components Standard




```
<custom-slider>
</custom-slider>
```

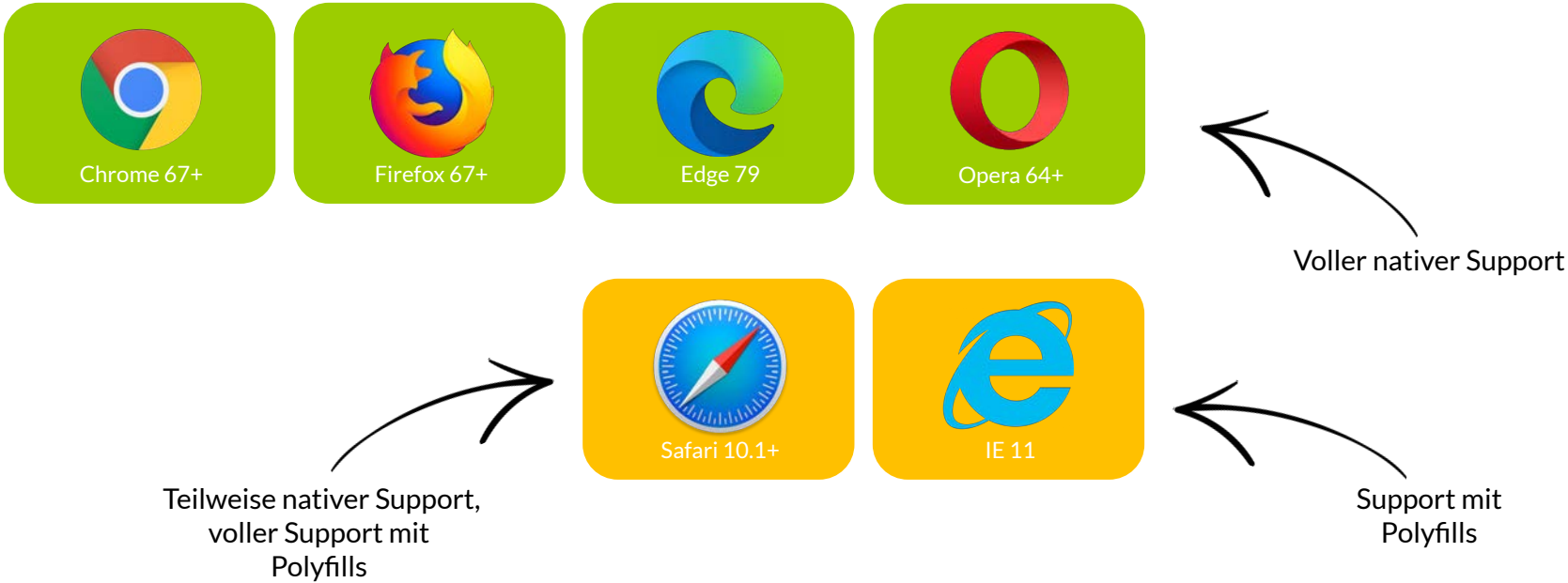
```
<custom-slider>
  #shadow-root(open)
    <style>…</style>
    <p>Shadow</p>
</custom-slider>
```

```
<template>
  <h1>Slider</h1>
</template>
```

```
import {CustomSlider}
from './components.js'
```

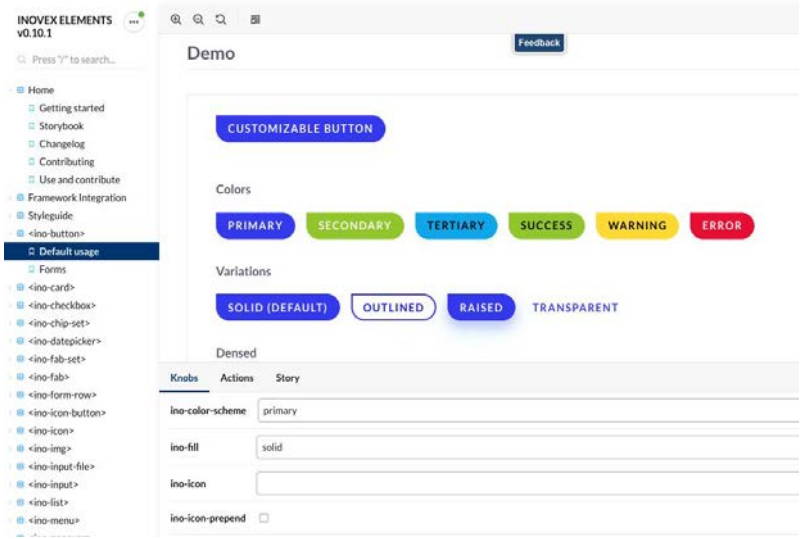
Erweiterung von HTML durch benutzerdefinierte, gekapselte und wiederverwendbare UI-Komponenten

Web Components Browser Support



Anwendungsszenarien

- Ergänzung von Web-Anwendungen
- Framework-übergreifende Komponenten-Wiederverwendung
- Micro Frontends
- Progressive Web Apps
- Design Systems

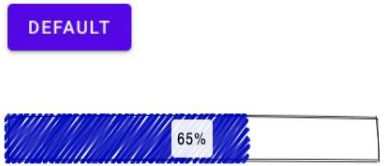


Entwicklung von Web Components

- Plain HTML, CSS, JavaScript
- Unterstützende Technologien
  - Reduzieren Code und Komplexität
  - Übernehmen das Data Binding
  - Integrierte Event Listener
  - Zusätzliche Lifecycle Methoden
- Framework-Komponenten Wrapper
  - Vue (@vue/web-component-wrapper)
  - Angular (@angular/elements)

Ready-to-use Web Component Libraries

- Material Web Components
- Polymer Elements
- Vaadin Components, Wired Elements, Elix, ...



inovex **elements**

Inovex Elements

Motivation

- Verringerung des Entwicklungsaufwands
- Erhöhung der Qualität
- Für heterogene Systeme (unterschiedliche Frameworks)

- Research-Plattform für neue Technologien
- Verbesserung Prozess (iterativ):
  - Design ⇒ Umsetzung ⇒ Integration ⇒ Design

inovex Elements

Eine weitere Bibliothek?

- Vorbehalte existierender UI-Bibliotheken
  - Beschränkung auf vorhandene Komponenten und Designentscheidungen
- Aber: Warum das Rad neu erfinden (z. B. Formulare, Datepicker, ...)

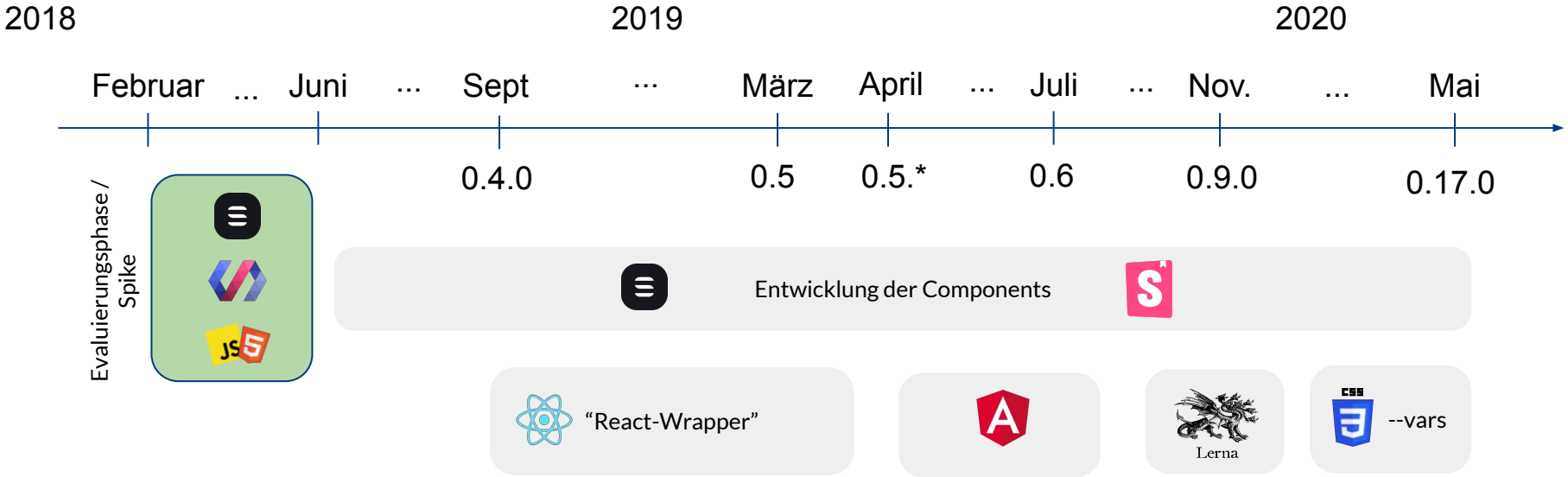
- Inovex Elements als “intelligenter Rahmen”
  - Bibliotheken sind ersetzbar und werden ein Implementierungsdetail
  - Elemente sind standardisiert.

Aktueller Stand: Featureset

- 39 inovex Components auf Basis von Stencil
  - Vollständig dokumentiert
  - Live-Playground via Storybook
  - Styleguide

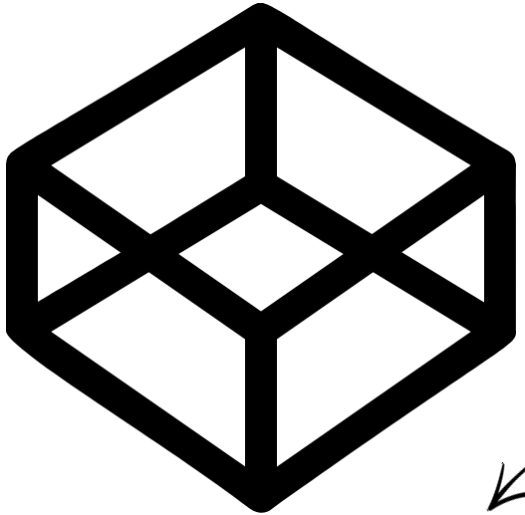
- Integration in React, Angular
- Continious Deployment auf elements.inovex.io

Milestones im Projektverlauf



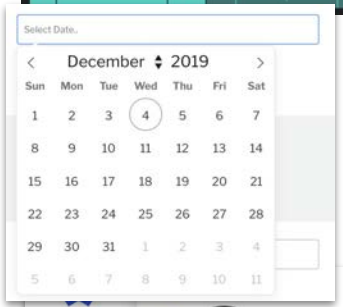
Getting Started

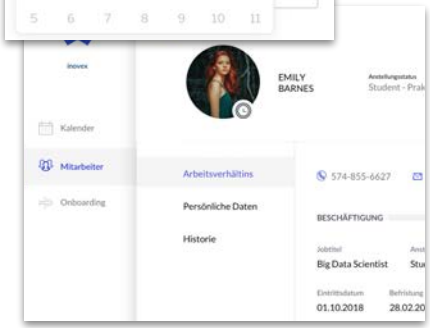
Demo



https://codepen.io/janivo/pen/qBdmOOy

elements: Dependencies





Google
Material Design **Web**

flatpickr

stencil

JAMES Design

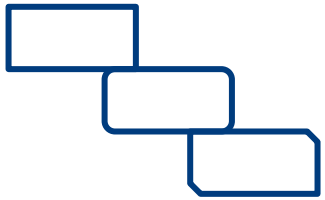
Storybook

Dokumentation

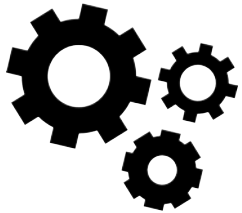
Google **Material** Design Web

Warum?

Google Material Design **Web**



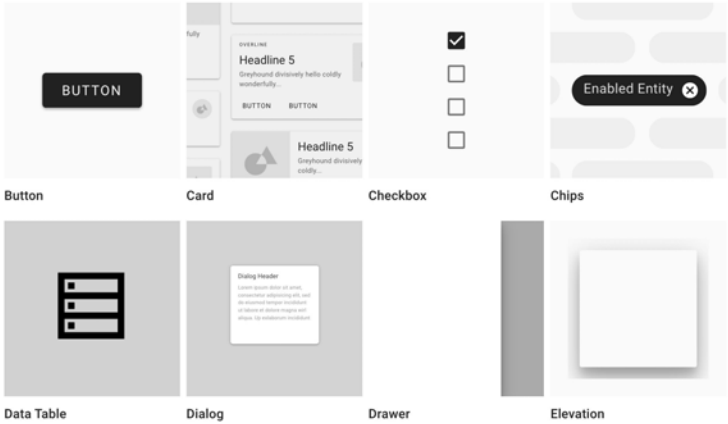
Primary		Secondary	
	#003C7E, Base		#9CCD00, Base
	#004A9C, Light		#BFF020, Light
	#002E62, Dark		#2E7D32, Dark



Google
Material Design **Web**

Klick

https://material-components.github.io/material-components-web-catalog



- mdc-backdrop
- mdc-banner
- mdc-base
- mdc-bottom-app-bar
- mdc-bottom-navigation
- mdc-bottom-sheet
- mdc-button
- mdc-card
- mdc-checkbox
- mdc-chips
- mdc-data-table
- mdc-density
- mdc-dialog
- mdc-divider
- mdc-dom
- mdc-drawer
- mdc-elevation
- mdc-fab
- mdc-feature-targeting
- mdc-floating-label
- mdc-form-field

https://github.com/material-components/material-components-web/tree/master/packages

Google
Material Design **Web**

npm

```
npm install @material/textfield
```

CSS

```
@import "@material/textfield/mdc-text-field";
```

HTML

```
<div class="mdc-text-field">
  <input type="text" id="my-text-field" class="mdc-text-field__input">
  <label class="mdc-floating-label" for="my-text-field">Label</label>
  <div class="mdc-line-ripple"></div>
</div>
```

JavaScript

```
import {MDCTextField} from '@material/textfield/index';
const textField = new MDCTextField(document.querySelector('.mdc-text-field'));
```

Stencil Component

stencil

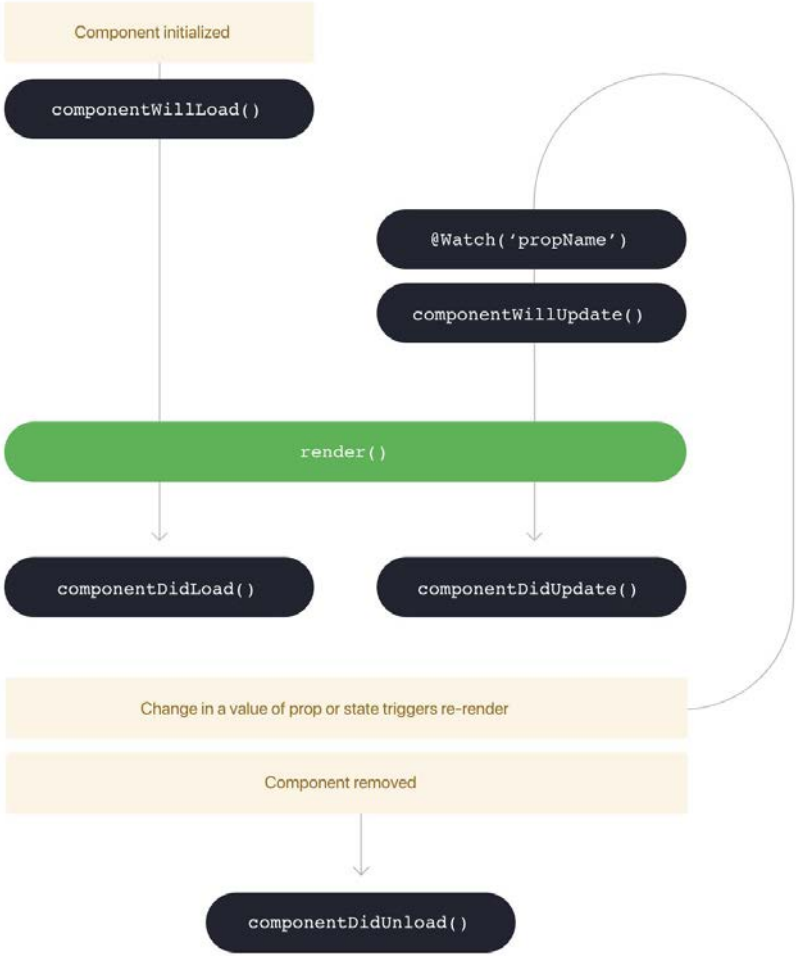
API

```
@Component({
  tag: 'ino-textarea',
  styleUrl: 'ino-textarea.scss',
  shadow: false
})
export class Textarea implements ComponentInterface {

  @Element() el!: HTMLElement;

  @Prop() value?: string = '';

  @Watch('value')
  handleChange(value: string) {/* */}

  @Event() valueChange!: EventEmitter<string>;

  @Listen('change')
  handleInput(e) { /* */ }

  componentDidLoad() { /* */ }
  componentWillUnLoad() { /* */}

  render() { /* */ }
}
```

https://stenciljs.com/

stencil

Lifecycle



Stencil +
Material Components

Wie sieht eine elements-Komponente aus?

```
@Component({
  tag: 'ino-icon-button',
  styleUrl: 'ino-icon-button.scss',
  shadow: false
})
export class IconButton implements ComponentInterface {

  // An internal instance of the icon button.
  private mdcInstance: MDCRipple;

  @Element() el!: HTMLElement;

  /**
   * Sets the autofocus for this element.
   */
  @Prop() autofocus?: boolean;

  /**
   * Disables this element.
   */
  @Prop() disabled?: boolean;
```

```
<ino-icon-button disabled="true">
</ino-icon-button>
```

stencil +

Wie sieht eine elements-Komponente aus?

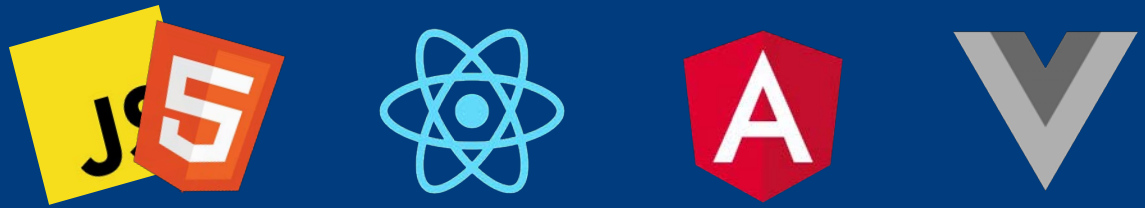
```
componentDidLoad() {
  const nativeElement = this.el.querySelector('.mdc-icon-button');
  this.mdcInstance = new MDCRipple(nativeElement);
}

componentWillUnload() {
  this.mdcInstance.destroy();
}

render() {
  return (
    <Host>
      <button class={'mdc-icon-button'} autofocus={this.autofocus} disabled={this.disabled}>
        <ino-icon ino-icon={this.inoIcon} class="mdc-icon-button__icon" />
      </button>
    </Host>
  );
}
```

MDC-Instanz erzeugen

MDC-Instanz zerstören

stencil +

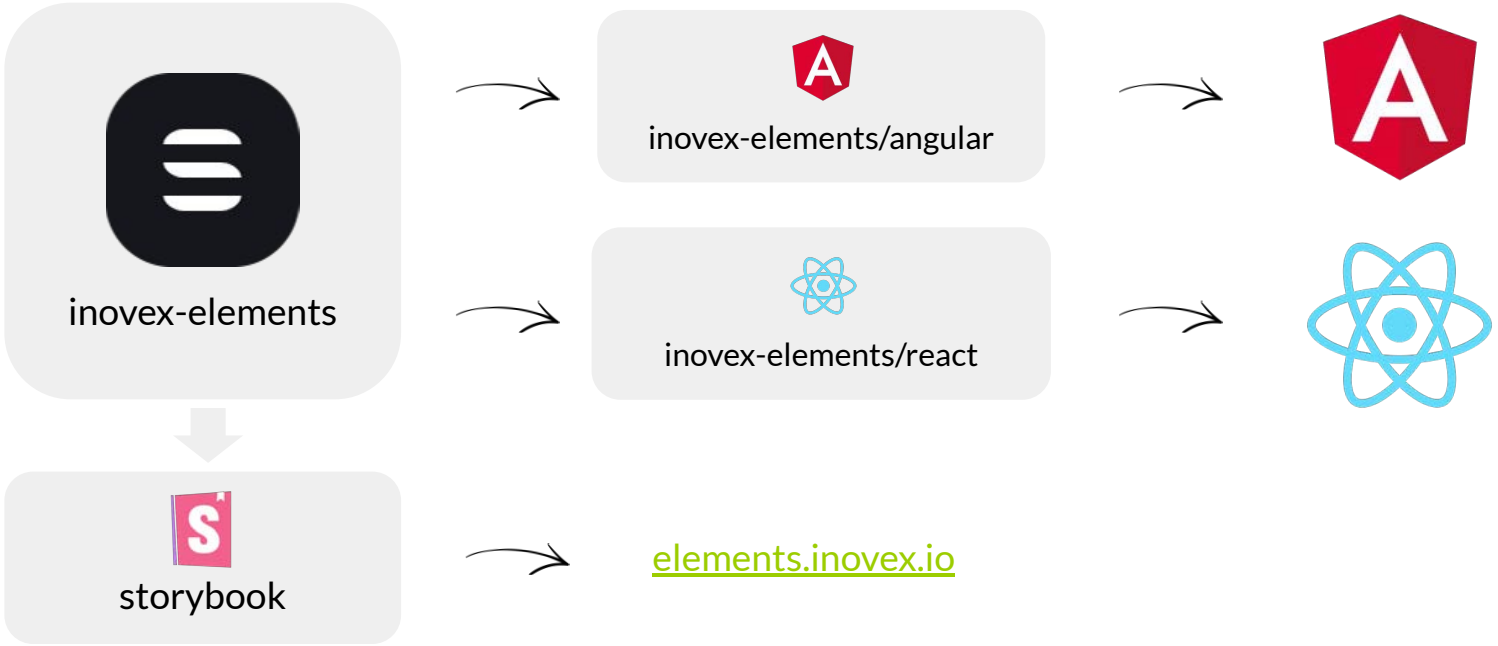
Framework **Integration**



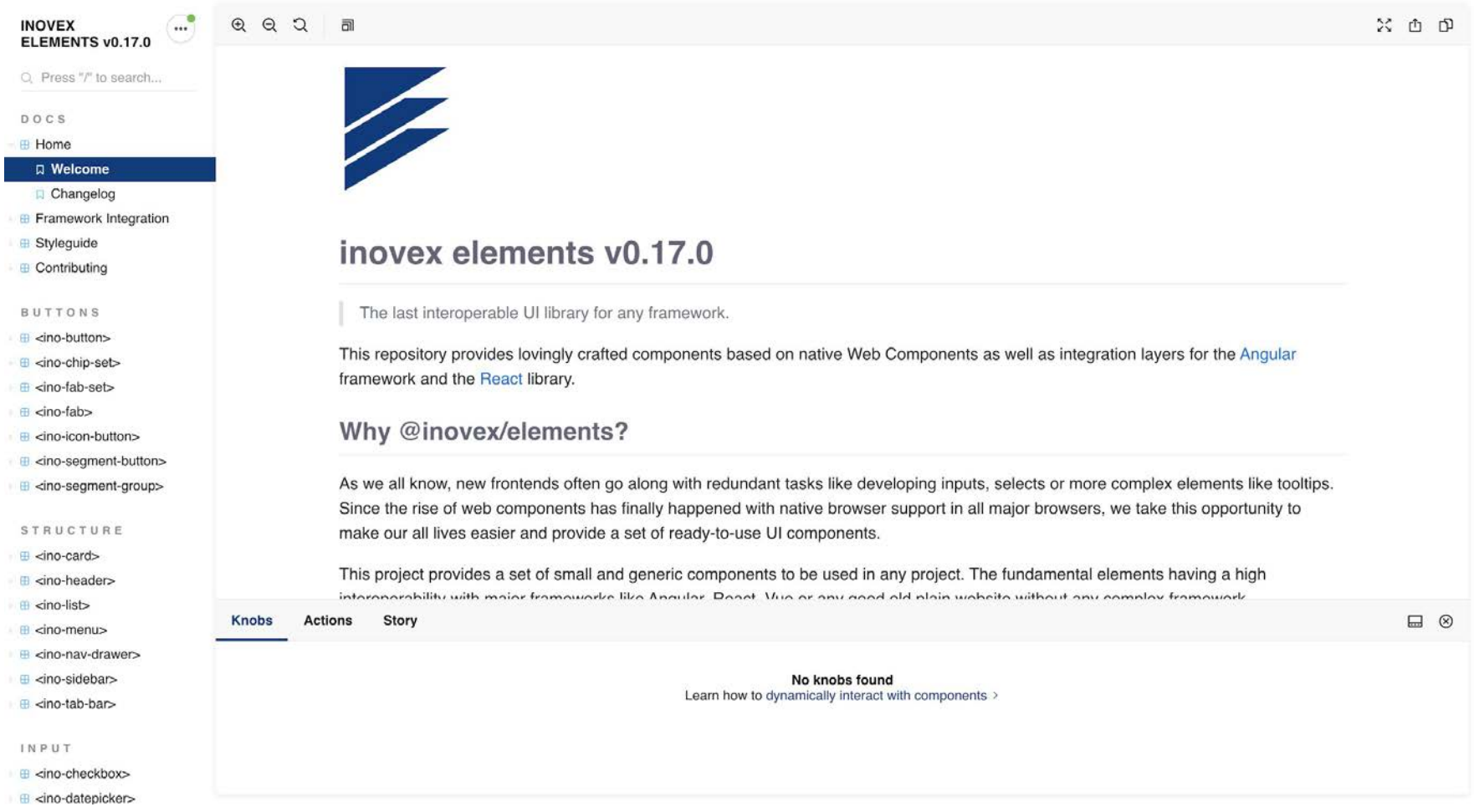
custom-elements-everywhere.com

Elements Packages

Monorepo



UI-Showcase

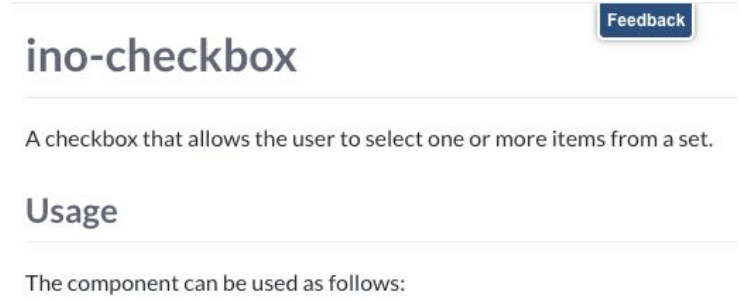

INOVEX ELEMENTS v0.17.0
Press "/" to search...
DOCS
Home
Welcome
Changelog
Framework Integration
Styleguide
Contributing
BUTTONS
<ino-button>
<ino-chip-set>
<ino-fab-set>
<ino-fab>
<ino-icon-button>
<ino-segment-button>
<ino-segment-group>
STRUCTURE
<ino-card>
<ino-header>
<ino-list>
<ino-menu>
<ino-nav-drawer>
<ino-sidebar>
<ino-tab-bar>
INPUT
<ino-checkbox>
<ino-datepicker>
inovex elements v0.17.0
The last interoperable UI library for any framework.
This repository provides lovingly crafted components based on native Web Components as well as integration layers for the Angular framework and the React library.
Why @inovex/elements?
As we all know, new frontends often go along with redundant tasks like developing inputs, selects or more complex elements like tooltips. Since the rise of web components has finally happened with native browser support in all major browsers, we take this opportunity to make our all lives easier and provide a set of ready-to-use UI components.
This project provides a set of small and generic components to be used in any project. The fundamental elements having a high
Knobs
Actions
Story
No knobs found
Learn how to dynamically interact with components ›

UI-Showcase

Komponenten-Dokumentation

```
# ino-checkbox
A checkbox that allows the user to select
one or more items from a set.

## Usage
The component can be used as follows:
...
```

Eigene Doku




```
/**
 * Marks this element as checked.
(**unmanaged**)
 */
@Prop() checked? = false;
```

von Stencil generiert

Property	Attribute	Description	Type	Default
`checked`	`checked`	Marks this element as checked. (**unmanaged**)	`boolean`	`false`
`disabled`	`disabled`	Disables this element.	`boolean`	`undefined`
`indeterminate`	`indeterminate`	Marks this element as indeterminate (**unmanaged**)	`boolean`	`undefined`
`name`	`name`	The name of this element.	`string`	`undefined`
`value`	`value`	The value of this element.	`string`	`undefined`

UI-Showcase

Storybook

Feedback

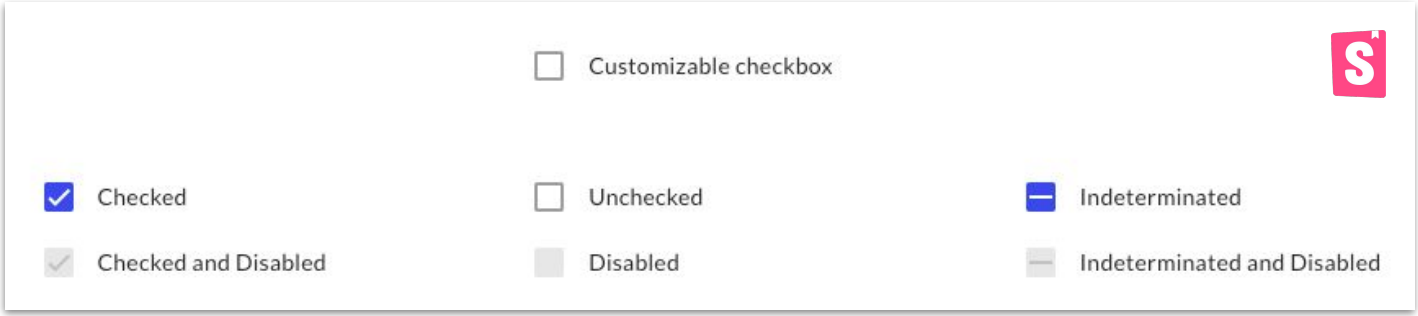
ino-checkbox

A checkbox that allows the user to select one or more items from a set.

Usage

The component can be used as follows:

Story →



Property	Attribute	Description	Type	Default
`checked`	`checked`	Marks this element as checked. (**unmanaged**)	`boolean`	`false`
`disabled`	`disabled`	Disables this element.	`boolean`	`undefined`
`indeterminate`	`indeterminate`	Marks this element as indeterminate (**unmanaged**)	`boolean`	`undefined`
`name`	`name`	The name of this element.	`string`	`undefined`
`value`	`value`	The value of this element.	`string`	`undefined`

Klick

elements.**inovex.io**

Lessons Learned

Lessons Learned

Web Components nicht überall sinnvoll

Integration von Web Components in Frameworks

Mono-Repository vereinfacht Entwicklung

```
packages/
├── elements/
├── elements-angular/
├── elements-react/
└── storybook/
```

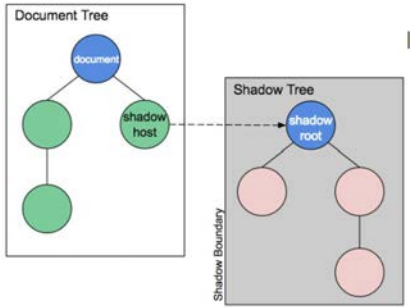
Lessons Learned

Zustandslose Komponenten erhöhen die Wiederverwendbarkeit

```
@State isOpen: boolean;
@Prop isOpen: boolean;
```

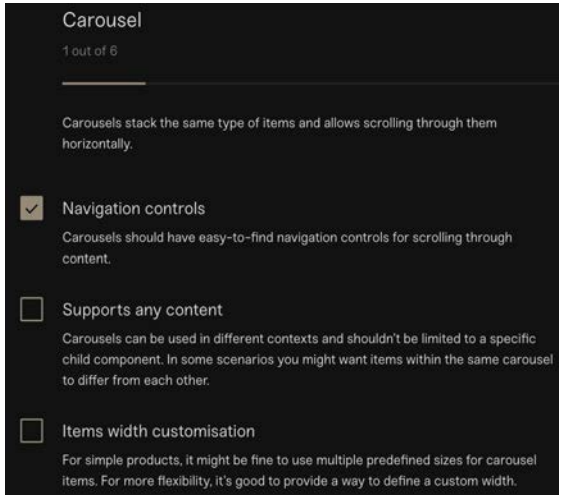
An anderen Bibliotheken orientieren hilft

Shadow-DOM schwer umsetzbar

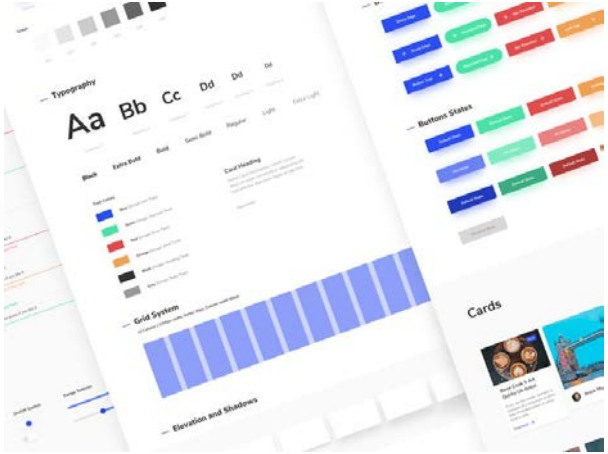


Next Steps

Next Steps



designsystemchecklist.com



Vielen Dank

Alessa Radkohl
aradkohl@inovex.de

Pascal Fecht
pfecht@inovex.de

Jan-Niklas Voß
jvoss@inovex.de

inovex GmbH
Ludwig-Erhard-Allee 6
76131 Karlsruhe

Rückblick: Erste Schritte 1/2

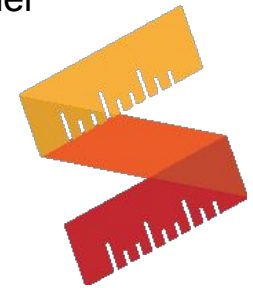
Evaluation: Welche Bibliothek ist überhaupt geeignet? (2018)

VanillaJS

StencilJS

Polymer

SkateJS



SlimJS

SWITZERLAND

Rückblick: Erste Schritte 2/2

Evaluation: Welche Bibliothek ist überhaupt geeignet? (2018)

Feature Map

Framework	Switzerland	slim.js	Stencil.js	Skate.js	Polymer 2	Polymer 3 (Rc)
Company	(Privat)	(Privat)	Ionic	Netlify	Google	Google
Aktive Entwickler	1	1	2-5	1	10-20	10-20
Stars	202	442	2623	2549	19336	19336
LOC	ca. 300	900	mittel	mittel	viel	viel
License	MIT	MIT	MIT	MIT	BSD Cluase	BSD 3 Clause
Community	Klein	Klein	Mittelgroß?	Klein	groß	groß
Dokumentation	wenig	wenig / mittel	mittel	wenig	viel aber unübersichtlich	viel aber unübersichtlich
Einschätzung: Geeignet für Enterprise	Nein	Nicht wirklich	Vielversprechend, aber noch alpha	Ja, aber mangelhafte Dokumentation	Ja, aber mit veralteten Standards wie HTML Imports	Eventuell, aber mit Breaking Changes
HTML-Imports	Nein	Nein	Nein	Nein	Ja	Nein
JSX	Ja	Ja	Ja	Ja	Nein	Nein, aber html-tags (lit-html?)
TypeScript?	Nein	Nein	Ja	Ja	Ja	Ja
Shadow DOM	Ja	Ja	Ja	Ja	Ja	Ja
Shadow DOM abschaltbar?	Nein	Ja	Ja	Ja	Mit Workaround	?