

# Howto



# Was erzählen wir euch heute?

Alles was ihr wissen müsst, um Vue zu benutzen und ein kleines Projekt auf die Beine zu stellen.

▶ Das alles in einer halben Stunde?

Hoffentlich.

▶ Cool.

Ja.

▶ Wer hat's erfunden?

▶ Inhaltsstoffe für unser nettes, kleines Projekt

▶ 2 Vues?!

▶ Was kommt in so ein „Vue“ rein?

▶ Was kann so ein „Vue“ machen?

▶ War das denn alles?

# Evan You



- ▶ Zuerst bei Google gearbeitet
  - ▶ Mit Angular.js
- ▶ Entwickelt Framework Vue & veröffentlicht es 2014

# Stack

## ▶ NPM

Package Manager, um Module einfach zu installieren und zu haushalten.

## ▶ Webpack & Babel

Server, der hot reloading ermöglicht und mit Babel ES6 transpiliert.

## ▶ Bulma & Font-awesome

CSS Framework (bzw. SASS) und Icon-Sammlung für schöne Websites.

## ▶ SASS

CSS-Präprozessor, weil wieso sollte man sich CSS antun.

## ▶ Vue

Duh.

# Standard vs. Components

Oder auch:

„Ich versteh nicht, was dir an Vue gefällt.“

vs.

„Aaaaah, das ist ja voll geil.“

# Standard

## ► Standard

```
new Vue({  
  el: '#app',  
  template: '<div>{{msg}}</div>',  
  props: ['msg']  
})
```

## ► CSS styles?

Müssen in extra .css Datei.

# Component.vue

```
<template lang="html">
  <div>
    {{msg}}
  </div>
</template>
```

```
<script>
export default {
  name: "component",
  props: ['msg`]
}
</script>
```

```
<style lang="css">
div::after
{ content: "Ich bin am selben Ort!"; }
</style>
```

# Instance Options



# Instance Options

- ▶ **Vue Standard Version**
  - ▶ Template
  - ▶ El
  - ▶ Data
  
- ▶ **Vue Component Version**
  - ▶ Components
  - ▶ Name
  - ▶ render
  - ▶ Data()
  - ▶ Methods
  - ▶ Computed
  - ▶ Props
  - ▶ Lifecycle-Functions

# Vue Standard Options

## ▶ El

Bietet der Vue-Instanz ein DOM-Element auf dem sie eingehängt werden kann.

Kann ein CSS-Selektor-String oder ein HTML-Element sein.

## ▶ Template

String Template ist ein Markup für die Vue-Instanz. Es ersetzt das durch el gegebene Element.

Wenn kein template angegeben, wird der Inhalt des el-Elements ausgelesen.

## ▶ Render

Alternative zu String Templates, bietet die Möglichkeit eine Javascript Funktion anstatt eines statischen Templates zu definieren.

Hier kann auch ein Component übergeben werden.

# Vue Component Options

## ► Components

Liste/Object von Components, welches in der Vue-Instanz verfügbar gemacht wird.

Können dann im Template als Tags verwendet werden.

## ► Name

Erlaubt dem Component sich selbst in seinem Template aufzurufen. Außerdem wird dieser Name automatisch als ID gesetzt, wenn die Instanz als Component verwendet wird.

Vue wird den Namen auch bei Fehlermeldungen anzeigen, anstatt „AnonymousComponent“.

## ► Data

Data Object für die Vue-Instanz. Seine Eigenschaften werden von Vue rekursiv in getters/setters convertiert, um es „reactive“ zu machen.

Diese Data ist auch im Template verfügbar.

Wenn in Components verwendet, muss data eine Funktion sein!

# Vue Component Options

## ▶ Props

Eine Liste/Object von Attributen, der es möglich ist data von ihrem Eltern-Component zu akzeptieren.

Erlaubt Spezifikation von type, default values, sowie ob die Prop required ist, und wann sie valide ist.

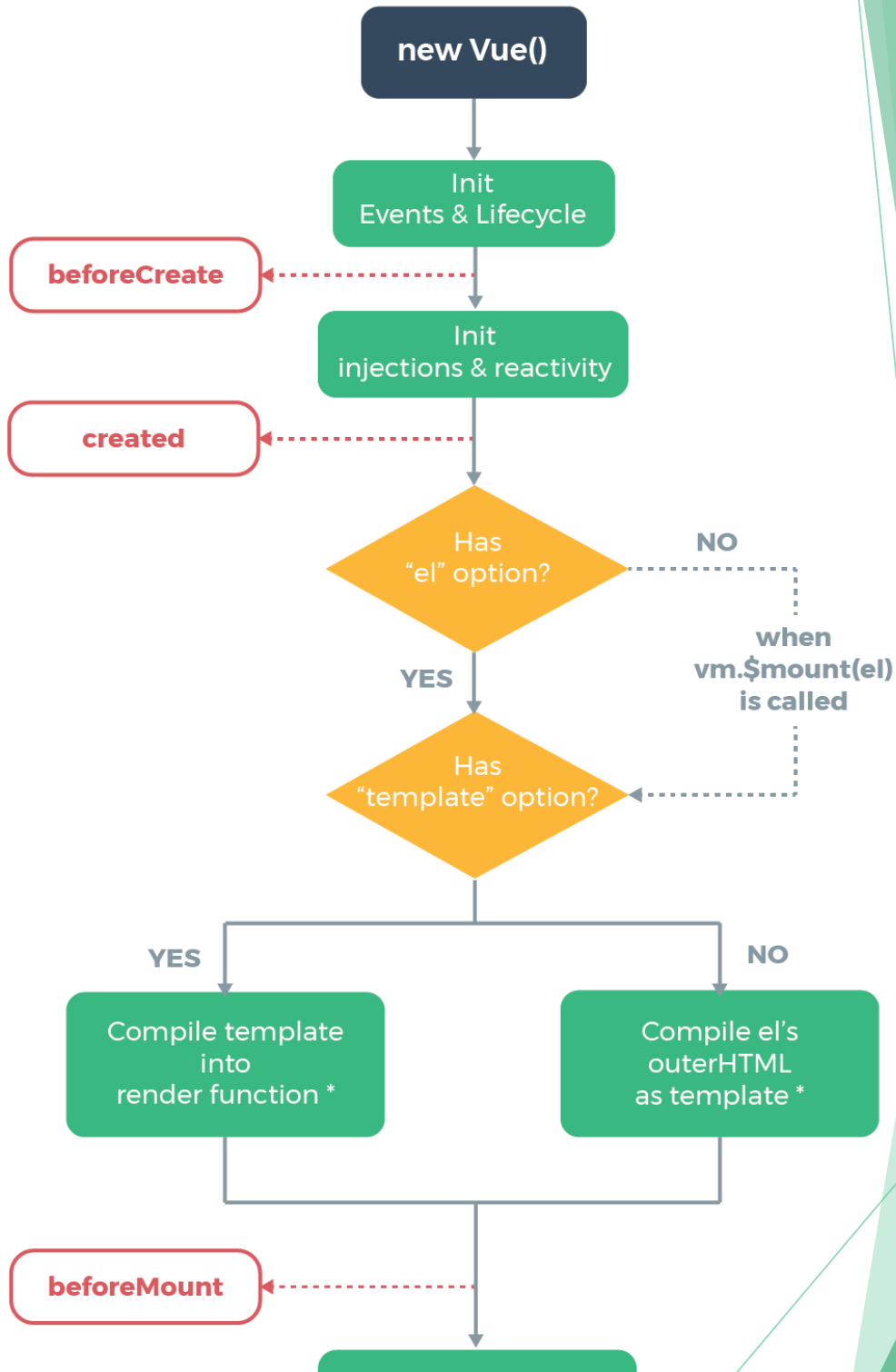
## ▶ Computed

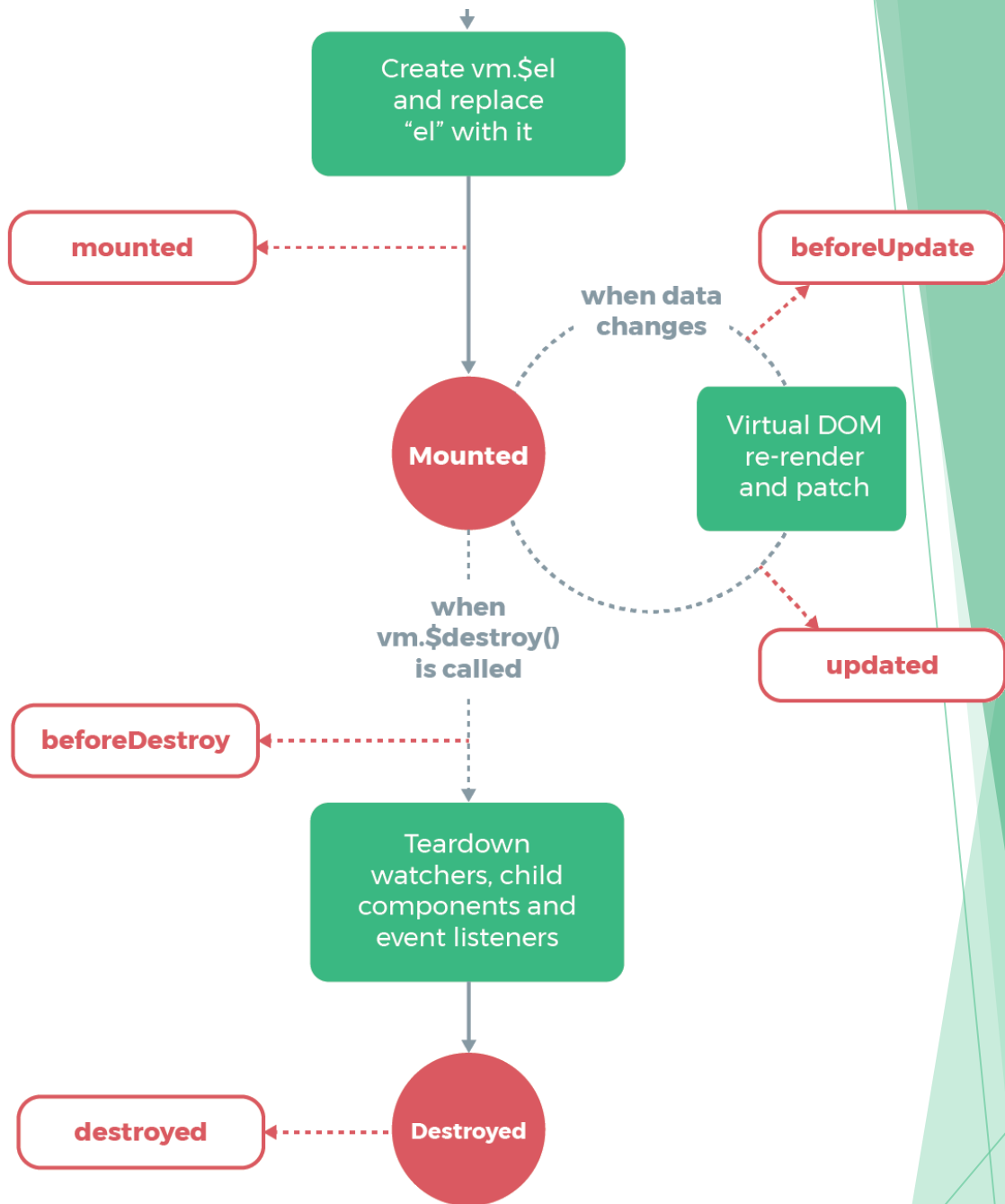
Computed Properties werden aus Data Properties erstellt. Sie werden gecached und nur neu berechnet, wenn die abhängige Data Property verändert wird.

## ▶ Methods

Deklariert Methoden, die im Template und innerhalb der Vue Instanz verfügbar gemacht werden.

# Lifecycle-Functions





\* template compilation is performed ahead-of-time if using a build step, e.g. single-file components

# Template Elemente und Direktiven

# Template Elemente und Direktiven

- ▶ V-if, v-else & v-show
- ▶ V-on & \$emit
- ▶ V-for
- ▶ V-bind
- ▶ slot
- ▶ V-model
- ▶ Transition
- ▶ Transition-group



# v-if, v-else & v-show

## ► Beschreibung

Rendert/zeigt das Element/Template je nach übergebenem booleschen Ausdruck.

## ► Besonderheiten

**v-if**: rendert und zerstört das Element bei Wechsel der Bedingung

**v-else**: wenn v-if nicht gerendert wird

**v-show**: rendert immer, zeigt aber nur bei wahren Ausdruck

## ► Code-Beispiel (App.vue)

```
<PostView v-if='loggedIn == true' ... />
```

```
<LoginView v-else ... />
```

# v-on & \$emit

## ► Beschreibung

v-on: Registriert Event-Listener auf das Element

**\$emit**: Feuert ein Event (Methode der Komponente)

## ► Besonderheiten

Shorthand: „@“

Modifiers: Viele, z.B. „prevent“: Verhindert, dass das Event weitergereicht wird (Seite wird bei submit Event nicht mehr neu geladen)

## ► Code-Beispiel (LoginView.vue)

```
<form @submit.prevent=' $emit ("login") '>  
<LoginView @login='loggedIn = true' ... />
```

# v-for

## ► Beschreibung

Rendert ein Element oder Template mehrmals, basierend auf einem Javascript-Array oder Object.

## ► Besonderheiten

Array: übergibt Element, Index

Object: übergibt Value, Key, Index

Key: benötigt, um gleichtypige Elemente zu unterscheiden

Statt „**alias in expression**“ geht auch „**alias of expression**“.

## ► Code-Beispiel (App.vue)

```
<Chirp v-for='chirp, index in chirps'  
:key='chirp.user + chirp.createdAt' ... >  
  ...  
</Chirp>
```

# v-bind

## ► Beschreibung

Bindet ein **Attribut/Prop** an eine Java-Variable.

## ► Besonderheiten

Shorthand: „:“

Rerender des Elements: automatisch bei Änderung der Variable! (Keine weiteren Listener notwendig, wie etwa bei Backbone.js)

## ► Code-Beispiel (App.vue)

```
<Chirp ... :user='chirp.user'  
:createdAt='chirp.createdAt'  
:likes='chirp.likes' ... >  
  ...  
</Chirp>
```

# slot

## ► Beschreibung

Wird automatisch durch Inhalt des Component-Tags ersetzt.

## ► Besonderheiten

Named oder **default**: können Namen zugewiesen bekommen, um mehrere slots zu erlauben

## ► Parent (App.vue)

```
<Chirp ... >  
  <strong slot="user">  
    {{chirp.user}}</strong>  
    {{chirp.text}}  
</Chirp>
```

## ► Child (Chirp.vue)

```
<slot name="user"></slot>  
  
...  
<slot></slot>
```

# v-model

## ► Beschreibung

Bindet eine Variable an den Value-Wert eines Textfelds. (Zwei-Wege-Bindung!)

## ► Besonderheiten

Modifizier: „**.trim**“ trimmt den Input

„**.number**“ castet zu Zahl

„**.lazy**“ reagiert auf Change event, statt Input

## ► Code-Beispiel (PostView.vue)

```
<textarea v-model.lazy.trim="text" ... >  
</textarea>
```

# transition

## ► Beschreibung

Realisiert eine CSS-Animation oder Transformation eines Elements.

## ► Besonderheiten

mode: „in-out“, „out-in“, default

## ► Im Template (App.vue)

```
<transition name='fade' mode='out-in'> ...  
</transition>
```

## ► Im Style (App.vue)

```
.fade-enter-active, .fade-leave-active  
{ transition: opacity 0.3s ease; }  
  
.fade-enter, .fade-leave-to  
{ opacity: 0; }  
  
.fade-enter-to, .fade-leave  
{ opacity: 1; }
```

# transition-group

## ► Beschreibung

Realisiert eine CSS-Animation oder Transformation mehrerer Elemente (z.B. v-for).

## ► Besonderheiten

Auch im DOM: wird als Element gerendert!

tag: gibt an, als welches Element die tg erscheint (default: span)

## ► Im Template (App.vue)

```
<transition-group name='bounce-in'  
tag='div' ...> ... </transition-group>
```

## ► Im Style (App.vue)

```
.bounce-in-enter-active  
{ animation: bounce-in-animation 0.5s; }  
.bounce-in-leave-active  
{ animation: bounce-in-animation 0.5s  
reverse; }  
.bounce-in-move  
{ transition: transform 0.5s;}
```



# Was haben wir (noch) nicht erwähnt?

## ▶ Vue-Router!

Es gibt ein npm Modul für Vue, Vue-Router, das Vue um eine Router-Funktionalität erweitert.

## ▶ Backend-Anbindung!

Vue.js hat die Entscheidung getroffen, auf eine Backend-Anbindung zu verzichten - es gäbe zu viele gute Alternativen von Dritten, um sich damit zu beschäftigen.

Häufig wird hier zu [Axios](#) gegriffen.

## ▶ Gaaaaaaaanz viel.

Wir haben nur die grundlegendsten Elemente von Vue gezeigt. Vue hat noch viel mehr zu bieten, wer Interesse hat, schaue sich in den Quellen um!

# Quellen

- ▶ Evan You

<https://i2.wp.com/wptavern.com/wp-content/uploads/2017/06/evan-you.png?fit=1200%2C621&ssl=1>

<https://medium.freecodecamp.org/between-the-wires-an-interview-with-vue-js-creator-evan-you-e383cbf57cc4>

- ▶ ...alles andere (für Leser):

<https://vuejs.org/v2/guide/>

<https://vuejs.org/v2/api/>

- ▶ Und (für Zuhörer):

<https://laracasts.com/series/learn-vue-2-step-by-step>