

Gin Gonic/ Vue.js

Presentation of the two stunning Webframeworks in
GoLang & JavaScript

by Alex Schübl, David Bochan, Nadin-Katrin Apel
in SS 2018



Table of Contents

1

Project setup: separation of backend & fronted

2

Why use a framework?

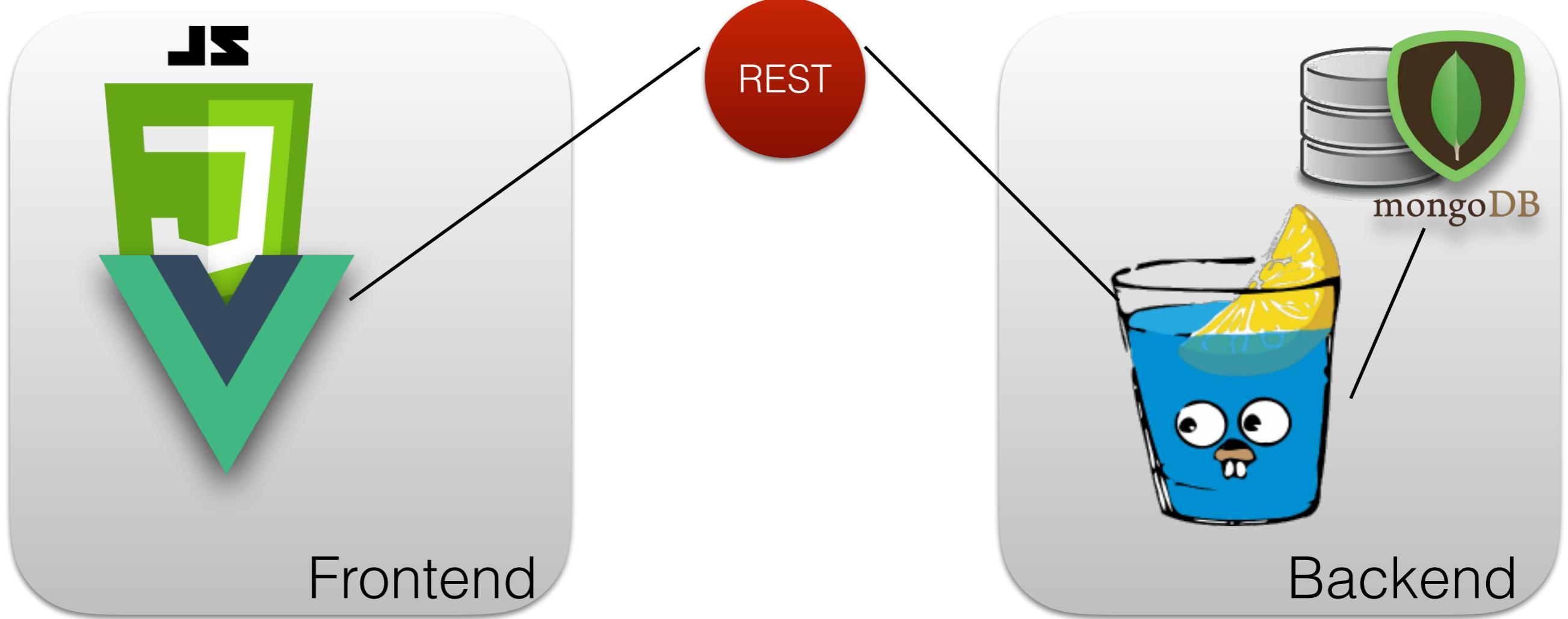
3

Main Concepts of Vue.js

4

Main Concepts of Gin Gonic

Separation of Backend/ Frontend



Recap from last week





HelloWorld ++

=

Calculator App



Table of Contents

1	The Vue Instance
2	Directives
2.1	Data Binding (<code>v-bind</code> , <code>v-model</code>)
2.2	Events (<code>v-on</code>)
2.3	Conditional Rendering (<code>v-show</code> , <code>v-if</code>)
2.4	Loops (<code>v-for</code>)
3	Components
4	The Vue CLI

Table of Contents

1	The Vue Instance
2	Directives
2.1	Data Binding (v-bind, v-model)
2.2	Events (v-on)
2.3	Conditional Rendering (v-show, v-if)
2.4	Loops (v-for)
3	Components
4	The Vue CLI

Get Started



```
<script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
```

Create a new Vue Instance

```
var app = new Vue({  
  el: '#app',  
  data: {  
    message: 'Hello Vue!'  
  }  
})
```

1

Get Started



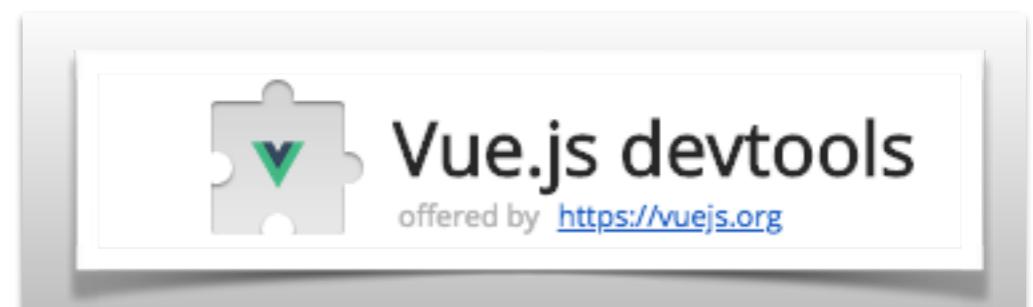
```
<script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
```

Exercise 1



- Create a new Vue Instance
- Install Vue developer tools as addOn in your browser
- create some data state attributes in the Vue instance
- play around in the browser console with your app data state in the dev tools & console

```
var app = new Vue({  
  el: '#calc-app',  
  data: {  
    name: 'Hero'  
  }  
})
```



Some Vocabulary



- Declarative Rendering
- Reactivity
- Interpolations
 - Text
 - Raw HTML
 - HTML attributes (eg. value, src)
- Directives

Table of Contents

1	The Vue Instance
2	Directives
2.1	Data Binding (<code>v-bind</code> , <code>v-model</code>)
2.2	Events (<code>v-on</code>)
2.3	Conditional Rendering (<code>v-show</code> , <code>v-if</code>)
2.4	Loops (<code>v-for</code>)
3	Components
4	The Vue CLI

Interpolation Basics

JavaScript Expressions

```
 {{name}}
 {{number + 1}}
 {{date ? 'yes' : 'no'}}
 <input type="text" v-on:input="input => name = input.target.value">
```

Directives

```
<input type="text" v-on:input="input => name = input.target.value">

<div v-html="rawHTML">
<input type="text" v-model="userInput">
```

class binding

```
<div class="static" :class="{ active: isActive }"></div>
<div :class="[activeClasses, errorClasses]"></div>
```

style binding

```
<div :style="{ width: points * 10 + '%', background: 'green' }"></div>
```



Shortages

v-on:input	@input
v-on:change	@change
v-on:click	@click
v-bind:value	:value
v-bind:src	:src

Interpolation Basics

The screenshot shows the WebStorm IDE interface. On the left, the code editor displays the `index.html` file of a Vue.js application. The code includes a simple text interpolation example:

```
<p>Some simple text interpolation with mustaches:</p>
<span class="bg-info">{{simpleText}}</span>
```

The right side of the interface shows a browser window titled "My First Vue App" with the URL `localhost:63342/awseomeVueApp/index.html`. The page content is: "Some simple text interpolation with mustaches: {{simpleText}}". Below the browser window is the Vue DevTools extension, which shows a component tree with a single node labeled "<Root>".

Interpolation Basics

JavaScript Expressions

```
 {{name}}
 {{number + 1}}
 {{date ? 'yes' : 'no'}}
 <input type="text" v-on:input="input => name = input.target.value">
```

Directives

```
<input type="text" v-on:input="input => name = input.target.value">

<div v-html="rawHTML">
<input type="text" v-model="userInput">
```

class binding

```
<div class="static" :class="{ active: isActive }"></div>
<div :class="[activeClasses, errorClasses]"></div>
```

style binding

```
<div :style="{ width: points * 10 + '%', background: 'green' }"></div>
```



Shortages

v-on:input	@input
v-on:change	@change
v-on:click	@click
v-bind:value	:value
v-bind:src	:src

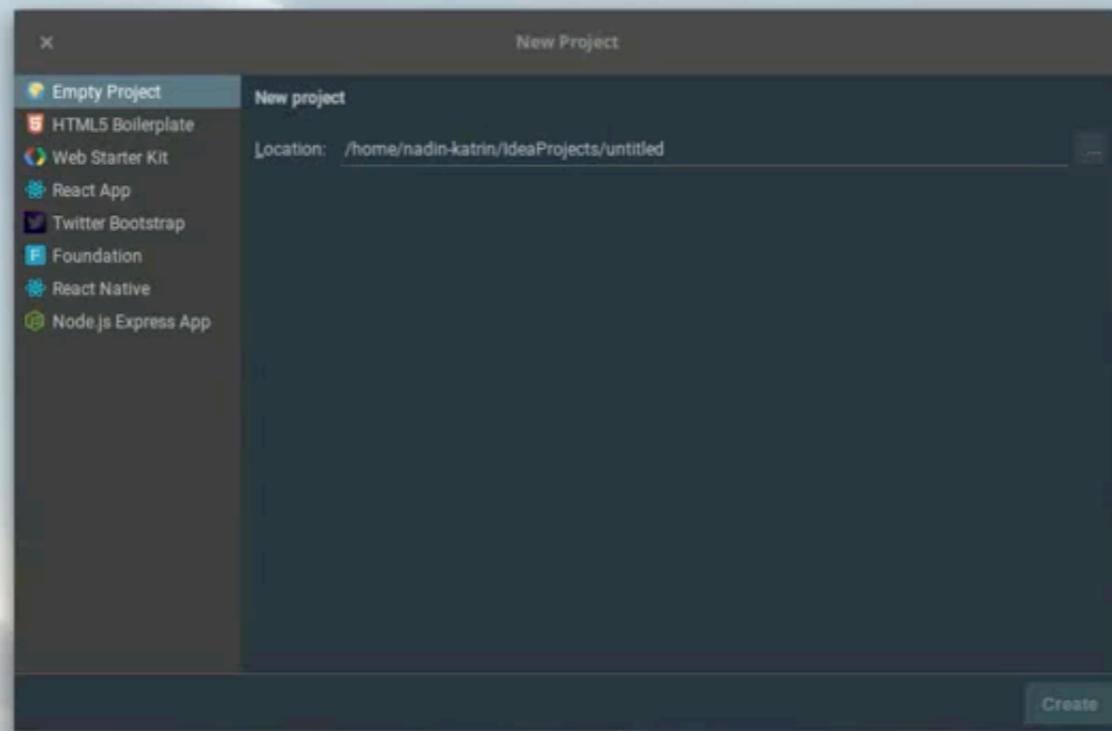
Exercise 2 a)

Become a Calculus Master

Calculation training completed at 5/3/2018 19:9:30 by Mathe Meister **Mathe Meister**

```
var getTimeStamp = function () {
    var today = new Date()
    return today.toLocaleDateString() + ' ' + today.getHours() + ':' + today.getMinutes() + ':' + today.getSeconds()
}
var randomNumberGenerator = function () {
    return parseInt(Math.random() * (99) + 1)
}
```

Exercise 2: Solution



Methods/ Events

```
<template>
  <input type="Number" v-model="a">
  <input type="Number" v-model="b">
  <button @click="result = a+b"></button>
  <input type="Number" v-bind:value="result">
</template>

<script>
  export default {
    name: "index",
    data() {
      return {
        a:23,
        b:234,
        result: null
      }
    },
  }
</script>
```



event: v-on:click = @click

1) inline method

Methods/ Events

event: v-on:click = @click

2) external method

```
<div id="app">
  <input type="Number" v-model="a">
  <input type="Number" v-model="b">
  <button @click="calculate()"></button>
  <input type="Number" v-bind:value="result">
</div>

<script>
  var app = new Vue({
    el: "#app",
    data() {
      return {
        a:23,
        b:234,
        result: null
      }
    },
    methods: {
      calculate(){
        this.result = this.a + this.b
      }
    }
  })
</script>
```

Methods

event: v-on:click = @click

2) external method
with parameters

```
<div id="app">
  <input type="Number" v-model="a">
  <input type="Number" v-model="b">
  <button @click="result = calculate(a,b)"></button>
  <input type="Number" v-bind:value="result">
</div>
<script>
  var app = new Vue({
    el: "#app",
    data() {
      return {
        a:23,
        b:234,
        result: null
      }
    },
    methods: {
      calculate(a,b){
        return a + b
      }
    }
  })
</script>
```

Exercise 2 b)

The screenshot shows a web application window titled "Become a Calculus Master". At the top, it displays a message: "Calculation training completed at 5/1/2018 21:05 by Mathe Meister" followed by a "Mathe Meister" badge. Below this is a horizontal input field containing the equation "60 + 11 =". To the right of the input field is a "New Game" button. The URL "https://youtu.be/9L3PLbnJ120" is visible at the bottom right of the window.

```
var getTimeStamp = function () {
    var today = new Date()
    return today.toLocaleDateString() + ' ' + today.getHours() + ':' + today.getMinutes() + ':' + today.getSeconds()
}
var randomNumberGenerator = function () {
    return parseInt(Math.random() * (99) + 1)
}
```

Exercise 2: Solution

The screenshot shows the WebStorm IDE interface with the 'calculator-app' project open. On the left, the 'index.html' file is displayed in the code editor:

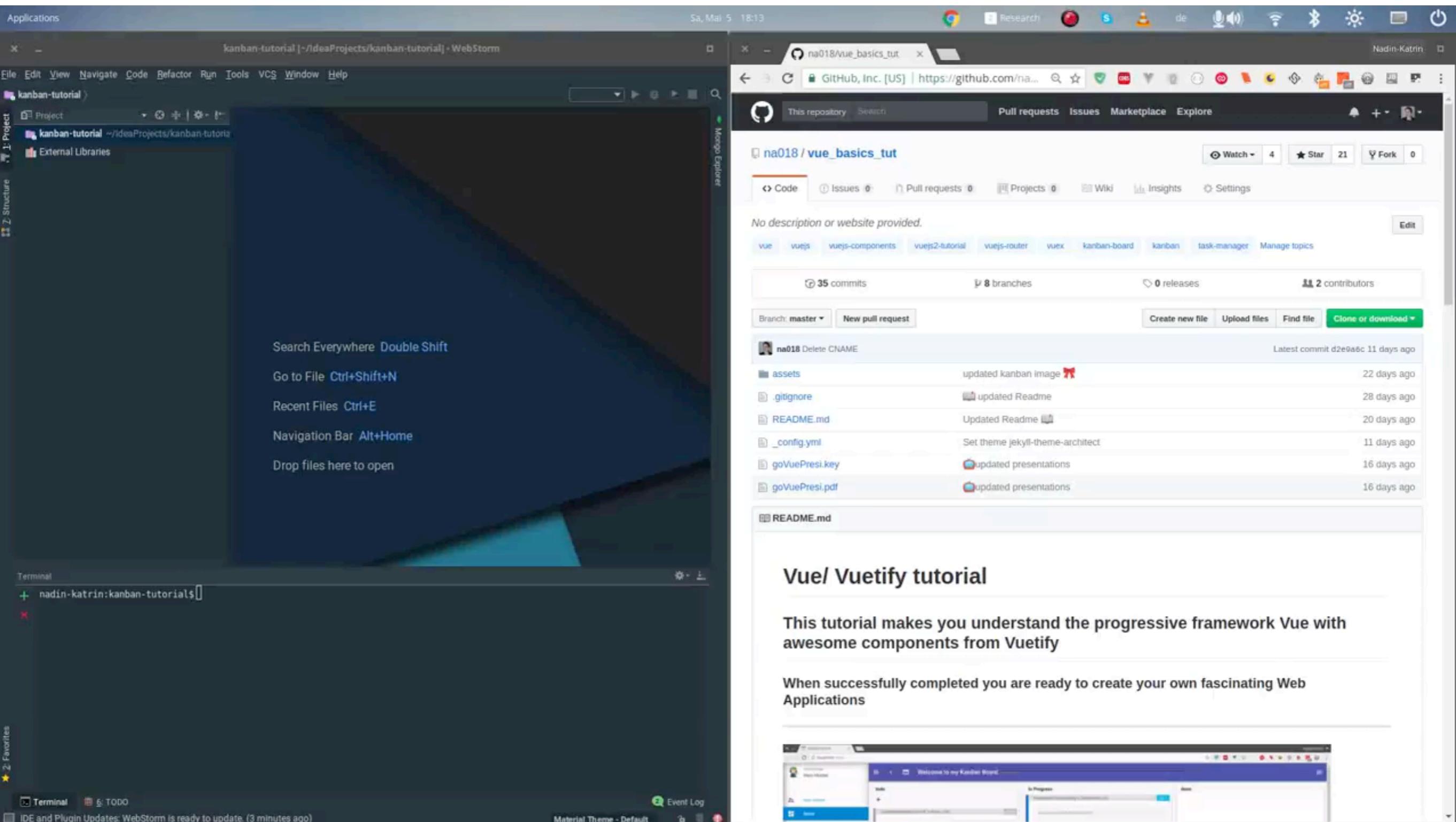
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Calculator App</title>
</head>
<body>
    <div id="rechenmeister-app">
        <h1>Become a Calculus Master</h1>
        <p>Calculation training completed at<br/>
            <span class="today">{{today}}</span><br/>
            by {{player}}<br/>
            <input type="text" v-model="player">
        </p>
    </div>
    <script src="https://cdn.jsdelivr.net/npm/vue@2.5.16/dist/vue.js"></script>
    <script>
        var getTimeStamp = function () {...}
        var app = new Vue({
            el: "#rechenmeister-app",
            data: {
                today: getTimeStamp(),
                player: 'Mathe Meister',
                points: 0
            }
        })
    </script>
</body>
</html>
```

On the right, a browser window titled 'Calculator App' shows the application running at `localhost:63342/calculator-app/index.html`. The page displays the text 'Become a Calculus Master' and 'Calculation training completed at' followed by the current date and time ('{{today}}') and the player's name ('{{player}}'). A text input field is present for entering the player's name, with the placeholder 'New Name 123'. Below the browser window, the Vue DevTools console shows the state of the application:

```
You are running Vue in development mode.  
Make sure to turn on production mode when deploying for production.  
See more tips at https://vues.org/guide/deployment.html
vue-devtools Detected Vue v2.5.16
> app.player
< undefined
> app.player="New Name"
< "New Name"
```

At the bottom of the IDE, the status bar indicates 'Unregistered VCS root detected: The directory /home/nadin-katrin is under Git, but is not registered... (20 minutes ago)'.

Recap: data binding kanban board



Conditional Rendering (v-if/ v-else, v-show)

```
<p v-if="date">Sven goes on a date with Juliet</p>
<p v-else-if="!date && practice">Sven goes to soccer practice</p>
<p v-else="!date && !practice">Sven watches Netflix lonely at home</p>
<p v-show="date">Sven is really happy :) </p>
```

Conditional Rendering (v-if/ v-else, v-show)

```
▼ parent.vue ×
1  <template>
2    <v-card class="pa-3">
3      <child-component name="Jessi" :age="4" place="New York"></child-component>
4      <child-component name="Jonas" :age="24" place="Stuttgart"></child-component>
5    </v-card>
6  </template>
```

Jessi is 4 years old and lives in New York

Our child has a very cute name! ❤️

Jonas is 24 years old and lives in Stuttgart

```
▼ child.vue ×
1  <template>
2    <div class="elevation-1 pa-1 ma-1">
3      <p>{{name}} is {{age}} years old and lives in {{place}}</p>
4      <p v-if="name === 'Jessi'">Our child has a very cute name! ❤️</p>
5    </div>
6  </template>
```

```
▼ child.vue ×
1  <template>
2    <div class="elevation-1 pa-1 ma-1">
3      <p>{{name}} is {{age}} years old and lives in {{place}}</p>
4      <p v-show="name === 'Jessi'">Our child has a very cute name! ❤️</p>
5    </div>
6  </template>
```

```
▼ <div class="elevation-1 pa-1 ma-1"> == $0
  <p>Jessi is 4 years old and lives in New York</p>
  <p>Our child has a very cute name! ❤️</p>
</div>
▼ <div class="elevation-1 pa-1 ma-1">
  <p>Jonas is 24 years old and lives in Stuttgart</p>
  <!---->
</div>
</div>
```

v-if

```
▼ <div class="elevation-1 pa-1 ma-1"> == $0
  <p>Jessi is 4 years old and lives in New York</p>
  <p>Our child has a very cute name! ❤️</p>
</div>
▼ <div class="elevation-1 pa-1 ma-1">
  <p>Jonas is 24 years old and lives in Stuttgart</p>
  <p style="display: none;">Our child has a very cute name! ❤️</p>
</div>
```

v-show

Become a Calculus Master

Calculation training completed at 5/3/2018 19:9:30 by Mathe Meister Mathe Meister

73 + 19 = 92

Exercise hints

```
<> head
  <> link https://stackp...otstrap/4.1.0/css/bootstrap.min.css
  <> meta
  <> title Calculator App
<> body
  <> div#rechenmeister-app
    <> div.header
      <> h1 Become a Calculus Master
    <> p Calculation training completed at by {{player}}
      <> span {{today}}
      <> input
    <> div.calculator {{op}}
      <> input
      <> input
      <> button =
      <> input
      <> button Test
      <> input
      <> p {{result === guessedResult ? 'yay – success' : 'oh...
      <> button New Game
<> script https://cdn.js...lvr.net/npm/vue@2.5.16/dist/vue.js
```

```
<> script
  & getTimeStamp()
  & randommNumberGenerator()
<> v app
  p "#rechenmeister-app"
  p el
  <> p data
    p today
    p player
    p points
    p a
    p b
    p result
    p guessedResult
    p op
<> p methods
  m recalculate(): void
```

Become a Calculus Master

Calculation training completed at 5/3/2018 19:9:30 by Mathe Meister Mathe Meister

73	+ 19	= 92	Test 92
day - success			
New Game			

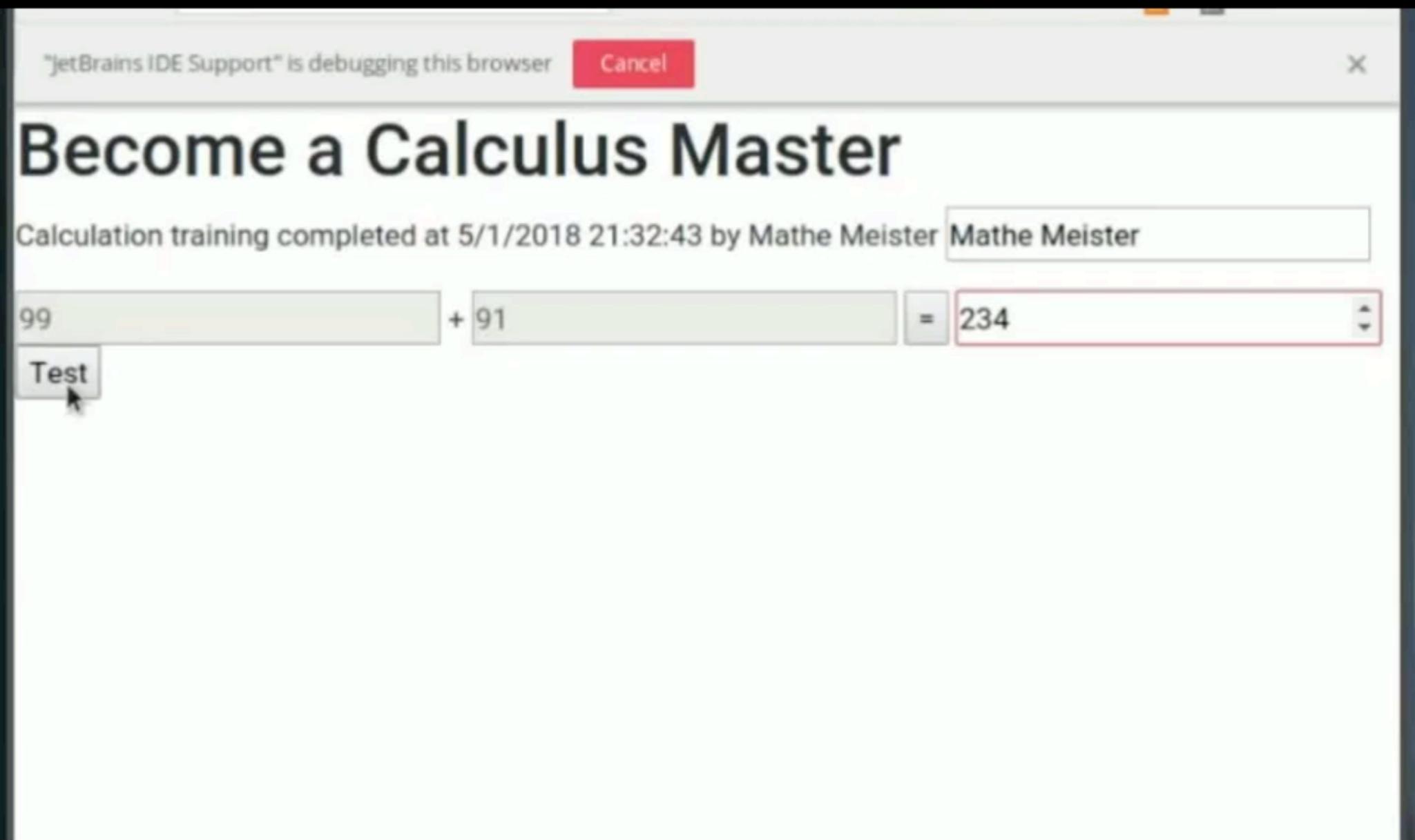
Exercise 2 c)

"JetBrains IDE Support" is debugging this browser Cancel X

Become a Calculus Master

Calculation training completed at 5/1/2018 21:32:43 by Mathe Meister Mathe Meister

99	+ 91	= 234
Test		



Exercise 2 c) Solution

The screenshot shows the WebStorm IDE interface with the 'calculator-app' project open. On the left, the 'index.html' file is displayed, containing Vue.js code for a calculator application. The code includes HTML templates, CSS classes, and JavaScript logic for calculating sums and managing player points. On the right, a browser window shows the running application at 'localhost:63342/calculator-app/index.html'. The page title is 'Calculator App'. The main content area displays the heading 'Become a Calculus Master' and a message indicating a calculation training completion at 5/1/2018 21:05 by 'Mathe Meister'. Below this, there is a simple calculator interface with input fields for numbers '25' and '92', an operator '+', and a result field. A 'New Game' button is also present. At the bottom of the browser window, the developer tools are open, specifically the 'Console' tab, which shows a message about running Vue in development mode and detected Vue v2.5.16.

```
<meta charset="UTF-8">
<ttitle>Calculator App</ttitle>
</head>
<body>
<div id="rechenmeister-app">
  <dtv class="header">
    <h1>Become a Calculus Master</h1>
    <p>Calculation training completed at
      <span class="today">{{today}}</span>
      by {{player}}
      <input type="text" v-model="player">
    </p>
  </dtv>
  <dtv class="calculator">
    <input type="number" :value="a" disabled> {{op}}
    <input type="number" :value="b" disabled>
    <button type="button" @click="result = a+b">=</button>
    <input type="number" v-model="guessedResult">
    <button type="button" @click="recalculate">New Game</button>
    <input type="number" :value="result" disabled>
  </dtv>
</div>
<script src="https://cdn.jsdelivr.net/npm/vue@2.5.16/dist/vue.js"></script>
<script>
  var getTimeStamp = function () {...}
  var randomNumberGenerator = function () {...}
  var app = new Vue({
    el: "#rechenmeister-app",
    data: {
      today: getTimeStamp(),
      player: 'Mathe Meister',
      points: 0,
      a: randomNumberGenerator(),
      b: randomNumberGenerator(),
      result: ''
    }
  })
</script>
```

2.4 Directives: loops (v-for)

```
<template>
  <li v-for="(player,index) in resultArray">{{player}}</li>
  <li v-for="(player,key, index) in resultObject" :key="player.key">{{player[key]}}</li>
</template>
```

```
data() {
  return {
    resultArray: [
      {name: 'hero', points: 97},
      {name: 'john', points: 123},
      {name: 'jess', points: 45},
    ],
    resultObject: {
      hero: {points: 97},
      john: {points: 123},
      jess: {points: 45},
    }
  },
},
```

Table of Contents

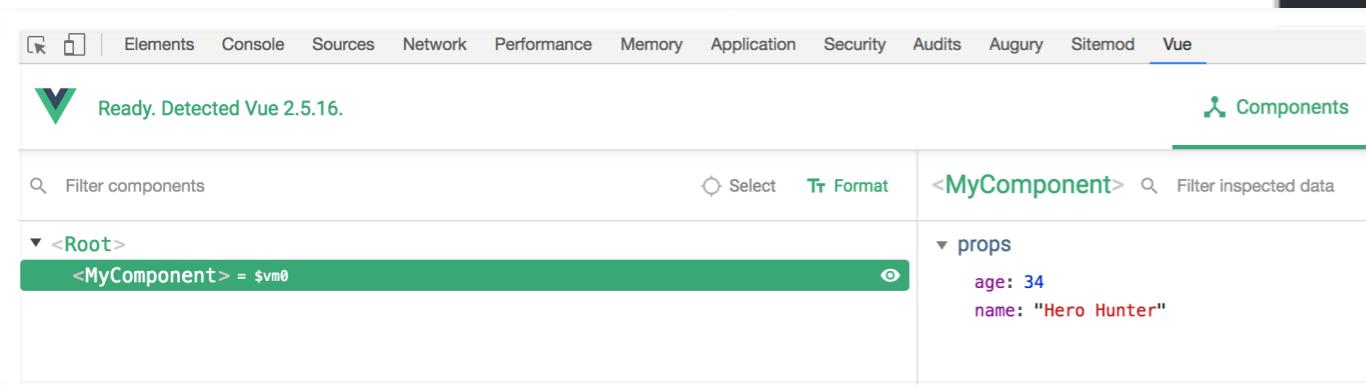
1	The Vue Instance
2	Directives
2.1	Data Binding (<code>v-bind</code> , <code>v-model</code>)
2.2	Events (<code>v-on</code>)
2.3	Conditional Rendering (<code>v-show</code> , <code>v-if</code>)
2.4	Loops (<code>v-for</code>)
3	Components
4	The Vue CLI

3. Components

```
<div id="app">
  <my-component :age="34" name="Hero Hunter"></my-component>
</div>
```

```
Vue.component('my-component', {
  props: ['age', 'name'],
  template: `<p>{{name}} is {{age}} years old</p>`,
})
```

```
Vue.component('my-component', {
  props: {
    age: {
      type: Number, required: false, default: 23},
    name: {
      type: String, required: true},
  },
  template: `<p>{{name}} is {{age}} years old</p>`,
```



3. Exercise Components

Become a Calculus Master

Calculation training completed at 5/4/2018 15:15:31 by Mathe Meister

Mathe Meister

2

+ 2

=

Test

Mathe Meister achieved 0 out of 10 on 5/4/2018 15:15:31

props: ['player', 'points', 'maxRounds', 'date']

3. Exercise Components: Solution

The screenshot illustrates the development environment for a web application. On the left, the WebStorm IDE shows the `index.html` file containing Vue.js code for a calculator application. On the right, a Google Chrome browser window displays the running application. The application's title is "Calculator App" and its main heading is "Become a Calculus Master". It shows a calculation: $63 + 66 = 129$. Below the calculation, a green success message "yay - success" is displayed. At the bottom, there is a "New Game" button.

```
</head>
<body>
<div id="rechenmeister-app">
  <div class="header">
    <h1>Become a Calculus Master</h1>
    <p>Calculation training completed at<br/>
      <span class="today">{{today}}</span>
      by {{player}}
      <input type="text" v-model="player">
    </p>
  </div>
  <div class="calculator">
    <input type="number" :value="a" disabled> {{op}}
    <input type="number" :value="b" disabled>
    <button type="button">>=</button>
    <input type="number" :value="guessedResult"
      @input="input => guessedResult = parseInt(input.target.value)" :disabled="result">
    <button type="button" @click="result = a+b">Test</button>
    <input v-if="result" type="number" :value="result" disabled>
    <p v-show="result" :class="result === guessedResult ? 'bg-success' : 'bg-danger'">{{result ===
guessedResult ? 'yay - success' : 'oh no - that was wrong'}}</p>
    <button v-show="result" type="button" @click="recalculate">New Game</button>
  </div>
</div>
<script src="https://cdn.jsdelivr.net/npm/vue@2.5.16/dist/vue.js"></script>
<script>
  var getTimeStamp = function () {
    var today = new Date()
    return today.toLocaleDateString() + ' ' + today.getHours() + ':' + today.getMinutes() + ':' +
    today.getSeconds()
  }
  var randomNumberGenerator = function () {
    return parseInt(Math.random() * (99) + 1)
  }
</script>
```

3. Exercise Components, loops, conditions

Become a Calculus Master

Calculation training completed at 5/4/2018 15:41:16 by Mathe Meister Mathe Meister

1/3

4 + 6 = I

3. Exercise Components, loops, conditions: solution

The screenshot shows a developer's environment with two main windows: a code editor and a browser developer tools interface.

Code Editor (WebStorm):

- Title Bar:** calculator-app [~/IdeaProjects/calculator-app] - ~/IdeaProjects/capp/index.html [calculator-app] - WebStorm
- Menu Bar:** File Edit View Navigate Code Refactor Run Tools VCS Window Help
- Project Tree:** Shows a project named "capp" with a file "index.html".
- Code Area:** Displays Vue.js code for a calculator application. The code includes a template with a progress bar and a script block defining a Vue instance with data properties like "player", "score", "rounds", and "date".
- Bottom Status Bar:** Unregistered VCS root detected: The directory /home/nadin-katrin is under Git, but is not register... (51 minutes ago) Material Theme - Default 38:1 LF: UTF-8

Browser DevTools (Chrome):

- Title Bar:** Calculator App
- Address Bar:** localhost:63342/calculator-app/Index.html
- Main Content:** "Become a Calculus Master" with a message: "Calculation training completed at 5/4/2018 15:15:31 by Mathe Meister Mathe Meister". Below it is a calculator interface showing "3 + 8 =". A progress bar indicates "Mathe Meister achieved 4 out of 10 on 5/4/2018 15:15:31".
- Network Tab:** Shows a single request to "localhost:63342/calculator-app/Index.html" with a status of "Success".
- Elements Tab:** Shows the DOM structure of the page.
- Console Tab:** Shows the output "Ready. Detected Vue 2.5.16."
- Sources Tab:** Shows the source code of the page.
- Performance Tab:** Shows the performance metrics of the page.
- Application Tab:** Shows the application state, including the Vue instance data:

 - <Root>**:
 - data**:
 - a: 3
 - b: 8
 - guessedResult: null
 - op: "+"
 - player: "Mathe Meister"
 - points: 4
 - result: null
 - today: "5/4/2018 15:15:31"

Recap: kanban components

The screenshot shows a desktop environment with two windows open. On the left is the WebStorm IDE interface for a project named 'kanban-tutorial'. The Project tool window shows a file structure with 'kanban-tutorial' at the root, containing 'vueTut' and '00_data_binding'. The terminal window shows a local session with the command 'PS1="\u:\w\\$' entered. On the right is a web browser window displaying a GitHub repository page for 'na018/vue_basics_tut'. The repository has 27 commits ahead of 'origin/exercis...' and 35 commits behind 'master'. The commit history lists various changes made by user 'david145', such as updating README.md, adding initial commits to build, config, server, src, static, and test files, and making improvements to .babelrc, .gitignore, .postcssrc.js, README.md, index.html, package-lock.json, package.json, and server.js. The commits are dated from 20 days ago to a month ago.

Search Everywhere Double Shift

Go to File Ctrl+Shift+N

Recent Files Ctrl+E

Navigation Bar Alt+Home

Drop files here to open

Local Local (1)

```
nadin-katrin@nadin-katrin-HP-Evy-17-Notebook:PC:~/IdeaProjects/kanban-tutorial$ PS1='\u:\w\$'
```

In this exercise you will learn about components, parent child relationships: how to pass data from the parent to the child & how to emit events in the child to tell the parent that something has happened

In the last tutorial we learnt about one-way and two-way data binding. One-way data-binding is accomplished with v-bind. With the help of this directive the Application data can be easily mapped to an HTML element (so *luckily no traversals through the HTML DOM are necessary for finding the intended element*) Two-way data-binding means we additionally listen to an event that could occur and change our application data (e.g. an input event on a text field). In vue this may easily be accomplished with v-model

Now have fun with this tutorial! 😊

Setup: Keep up & running

Recap: kanban loops

The screenshot shows a desktop environment with two windows open. The left window is the WebStorm IDE, showing the project structure for 'kanban-tutorial'. The right window is a web browser displaying a GitHub exercise titled 'Exercise 3: Directives (conditionals, loops)'. The exercise content discusses directives like v-if and v-for, and includes setup instructions and a terminal command for running the application.

Exercise 3: Directives (conditionals, loops)

In this exercise you will learn about directives with focus on conditional one (`v-if`) and loops (`v-for`). In the last tutorial you learned how to pass data from a parent to a child component. In this tutorial you will reuse the child component in another component to understand the benefits of reusable components. Additionally you will emit an event from a child component (`this.$emit('functionCalledInParent')`) to inform the parent component something has happened. (Here a user has clicked on a card). Now have fun with this tutorial! 😊

Setup: Keep up & running

```
# download branch (make sure you are one level above vueTut)
git clone -b origin/exercise/02_directives https://github.com/na018/vue_basics_tut.git vueTut/02_directives

#prerequisite: node.js is installed
node -v                                #returns for example v8.2.1

# install dependencies
npm i

#start the json mock server
node server.js

# run the application (should open a browser window automatically)
npm start
```

Result

ToDo

In src/components/userAdminComponents/ActivateUser.vue:

- [KB-3] use '`<user-card>`' for displaying the user nicely (reuse)

Table of Contents

1	The Vue Instance
2	Directives
2.1	Data Binding (<code>v-bind</code> , <code>v-model</code>)
2.2	Events (<code>v-on</code>)
2.3	Conditional Rendering (<code>v-show</code> , <code>v-if</code>)
2.4	Loops (<code>v-for</code>)
3	Components
4	The Vue CLI

The Vue CLI

The screenshot shows a development environment with two main windows. On the left is the WebStorm IDE interface, displaying the project structure and the content of the `index.html` file. The right window is a web browser showing a Vue.js application titled "My Vue Application". The application displays a calculation result: "88 + 26 = 114" and includes a "Test Result" button. A sidebar on the right features a cartoon illustration of a rabbit and the text "Rechenmeister".

index.html

```
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.0/css/bootstrap.min.css" integrity="sha384-9gVQdYFwwSjIDZnLEWnxCjeSWFphJlwGPXr1jddIhOegiu1Fw05qRGvFXOdJZ4" crossorigin="anonymous">
    <!-- <link rel="stylesheet" type="text/css" href="static/bootstrap.css">-->
    <link rel="stylesheet" type="text/css" href="static/css/main.css">
    <title>My Vue Application</title>
  </head>
  <body>
    <div id="rechenmeister-app" class="container"></div>
    <script src="https://cdn.jsdelivr.net/npm/vue@2.5.16/dist/vue.js"></script>
    <script>
      Vue.component('calculation-result', {
        props: ['player', 'score', 'rounds', 'date'],
        template:
          `<div class="calculation-result-row"><p>{{player}} achieved {{score}} out of {{rounds}} on {{date}}</p>
           <div class="point-indicator bg-danger">
             <p class="bg-success" :style="width: score * 100/rounds + '%'"></p>
           </div></div>`,
      })
      var randomNumberGenerator = function () {
        return parseInt(Math.random() * (99) + 1)
      }
      var getTimeStamp = function () {
        let today = new Date()
        return today.toLocaleDateString() + ' ' + today.getHours() + ':' + today.getMinutes() + ':' + today.getSeconds()
      }
    </script>
```

WS My Vue Application

localhost:63342/02_vue_tut_basic/Index.html

"JetBrains IDE Support" is debugging this browser

Cancel

Become a Calculus Master

Calculation training completed at 5/6/2018 14:20:5 by Mathe Meister

Round: 1/5

88 + 26 = Test Result

1 + 2 = 3

Rechenmeister

KanBan Board

Welcome to my KanBan Board

User Admin

Board

In Progress

done

todo

Implement UserCard view for displaying Users nicely

Understand Lenses (Projected) and parent-child component relationship

Understand directives w/ v-for

Accesibility update UserAdmin

Change the actions icon to Discard/Reset

If someone clicks on a UserCard by a component should emit an event containing the user id and the component name

Get all Users & active User from users store

understand getters w/ user store

Set active user globally

The navigation & kanban board should know the active user. Update the user globally in the user store

Understand routing

Implementation routes in "2" board. Home to Dashboard page & KanbanBoard page

Help the user to find their board

an item component has dependency leftHandLinks out, add link inc closed

Thanks for your Attention

Nadin-Katrin Apel, Alexander Schübl, David Bochan



You should now understand the basics of Vue.js