



kreativni  
*meetup*

# Full stack **2019 EDITION**

---

**MATJAŽ LIPUŠ**

## Agenda

---



- Web components
- Frameworks/libraries
- Vue.js
- Nuxt.js
- Testing
- GraphQL
- Hasura



## Web components

---



- Lifecycle events
- Templates
- Router
- State
- Mutation
- VDOM

## Element

---

Element, a Vue based component library for developers, designers and product managers.

[element.eleme.io](https://element.eleme.io)



**Basic controls,  
grid system,  
icons...**

**Combobox,  
Date picker,  
Table,  
Tree,  
Tabs,  
Steps,  
Carousel...**



# Pug

---



## Shorter HTML

- Case
- Conditions
- Filters
- Blocks
- Mixins
- Includes...

```
a.button(target="_blank") Click me
```

```
|
```

```
table.table
```

```
tr
```

```
td This
```

```
td are
```

```
td table
```

```
td columns
```

# {less}

---



## Leaner Style Sheets

- Variables
- Mixins
- Nesting
- Operators
- Functions
- Maps...

```
@width: 10px;
#header {
  color: black;
  .navigation {
    font-size: 12px;
  }
  .logo {
    width: @width;
  }
}
```

# TypeScript

---



JavaScript that scales.

- JavaScript types superset
- Types
- Getter/Setters
- Decorators
- Generics
- ...





## The Progressive Framework

- Lifecycle events
- Templates
- Router
- State
- Mutation
- VDOM

```
.vue
|
|
|<div id="app-5">
|  <p>{{ message }}></p>
|  <button
|    @click="reverseMessage">Reverse
|    Message</button>
|</div>
|
|
```





# The Progressive Framework

- Lifecycle events
- Templates
- Router

- State
- Mutation
- VDOM

## .vue - single file components

---



```
<template>
  <div id="app">
    <p>{{ message }}</p>
    <button @click="reverseMessage">Reverse Message</button>
  </div>
</template>

<script>
var app5 = new Vue({
  el: '#app',
  data: {
    message: 'Hello Vue.js!'
  },
  methods: {
    reverseMessage: function () {
      this.message = this.message.split('').reverse().join('')
    }
  }
})
</script>
```

Nuxt.js

---



# The Vue.js Framework

for

- Mobile Applications
- Desktop Applications
- Universal Applications

- Static Generated Applications
- Single Page Applications (SPA)
- Progressive Web Apps (PWA)

# Nuxt.js

---



- Code splitting
- Router
- SSR
- ES6/7, TypeScript(ts)
- Managing <head>
- Pre-processors (less, pug, ts)
- HTTP/2 push headers
- extensible

- Configuration
- Routing
- Pages
- async data
- Assets
- Plugins
- Modules
- State

# Testing

---



Provides information about the quality of the software.

## Correctness

Program is expected to be doing what it should.

## Maintainability

Provides feedback for easier software changes.

# Unit testing



```
<template>
  <span>{{ message }}</span>
</template>

<script>
  export default {
    data () {
      return {
        message: 'hello!'
      }
    },
    created () {
      this.message = 'bye!'
    }
  }
</script>
```

```
describe('MyComponent', () => {
  // Evaluate the results of functions in the raw
  component options
  it('sets the correct default data', () => {
    expect(typeof MyComponent.data).toBe('function')
    const defaultData = MyComponent.data()
    expect(defaultData.message).toBe('hello!')
  })

  // Inspect the component instance on mount
  it('correctly sets the message when created', ()
=> {
    const vm = new Vue(MyComponent).$mount()
    expect(vm.message).toBe('bye!')
  })

  // Mount an instance and inspect the render output
  it('renders the correct message', () => {
    const Constructor = Vue.extend(MyComponent)
    const vm = new Constructor().$mount()
    expect(vm.$el.textContent).toBe('bye!')
  })
})
```

## End to end testing

---



Test from start to finish

- Simulates events
- Assert state
- Screenshots
- Network

[Nuxt.js End-to-End Testing](#)

[E2E Testing with Nightwatch](#)

# Gherkin

---



Describe business behavior without the need to go into detail of implementation.

- Feature
- Background
- Scenario
- Given
- When
- Then

## Feature: Guess the word

# The first example has two steps

### Scenario: Maker starts a game

When the Maker starts a game

Then the Maker waits for a Breaker to join

# The second example has three steps

### Scenario: Breaker joins a game

Given the Maker started a game with the word "silky"

When the Breaker joins the Maker's game

Then the Breaker must guess a word with 5 chars

[What is Gherkin?](#) [Gherkin site](#)



# GraphQL

---



Data query and manipulation language for APIs,  
and a runtime for fulfilling queries with existing data.

- Ask for what you need, get exactly that
- Get many resources in a single request
- Describe what's possible with a type system

- Evolve your API without versions
- Great dev. Tools (cli, GraphiQL)
- Bring your own data and code

# GraphQL

---



- Protocol agnostic
- Fields
- Resolvers
- Schema
- Queries
- Mutations
- Subscriptions

```
query {  
  person {  
    name  
  }  
}
```

## 3factor app

---



Architecture pattern for modern full-stack apps.

- Realtime GraphQL
- Reliable eventing
- Async serverless

- Dynamic access control & auth
- Trigger webhooks on database events
- Remote Schemas



## Instant Realtime GraphQL on Postgres.

- GraphQL queries & mutations
- Realtime with subscriptions & live-queries
- Event-triggers on database events

- Dynamic access control & auth
- Trigger webhooks on database events
- Remote Schemas

# Nuxt.js ♥ Hasura

---



Connected automatically.

- Vue.js
- GraphQL
- Apollo
- Hasura

[Example](#)



# Thanks!

Questions, debate & pizzas