



EFFECTIVENESS  
EFFICIENCY  
TRANSPARENT

ENTERPRISE ARCHITECTURE  
EXCELLENCE

ARC  
INNOVATIONS

START  
HERE...

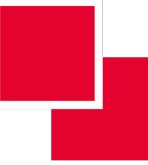
# {DEV}elopers {K}öln Meetup - „GWT“

Google Web Toolkit

Hossfeld:  
Solutions GmbH



**Frank Hossfeld**  
**Lukas Chrissty-Schmidt**  
03.07.2019



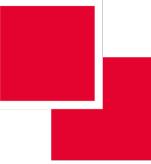
## Lukas Chrissty-Schmidt

- Mehr als 10 Jahre Erfahrung im IT-Versicherungsumfeld
- Gesellschafter bei arc innovations
- Unterstützt als Solution-, Businessarchitect, Projektmanager und Managementberater mit Fokus auf das operative und strategische Architekturmanagement.
- Kontakt: lcs@arc-innovations.de



## arc innovations GmbH & Co. KG:

- Unabhängiges Beratungshaus
- Spezialisiert auf die Versicherungsbranche
- Ganzheitliches Beratungskonzept: Von der Strategie bis hin zur Umsetzung
- [www.arc-innovations.de](http://www.arc-innovations.de)



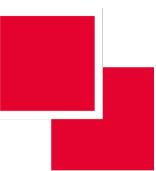
## Frank Hossfeld

- Seit über 30 Jahren in der IT unterwegs
- Geschäftsführender Gesellschafter der Hossfeld Solutions GmbH
- Unterstützt als Solution-Architekt, Frontend- und GWT-Spezialist die strategische Umsetzung von Frontend-Projekten
- Kontakt: [info@hossfeld-solutions.de](mailto:info@hossfeld-solutions.de)



## Hossfeld Solutions GmbH:

- Unabhängiges Systemhaus
- Consulting, Schulungen, Projektarbeit, Gewerke
- [www.hossfeld-solutions.de](http://www.hossfeld-solutions.de)



## Agenda

**Was ist GWT und seine Historie**

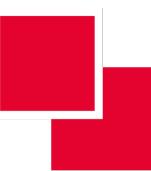
**GWT heute**

**GWT Frameworks und seine Integration**

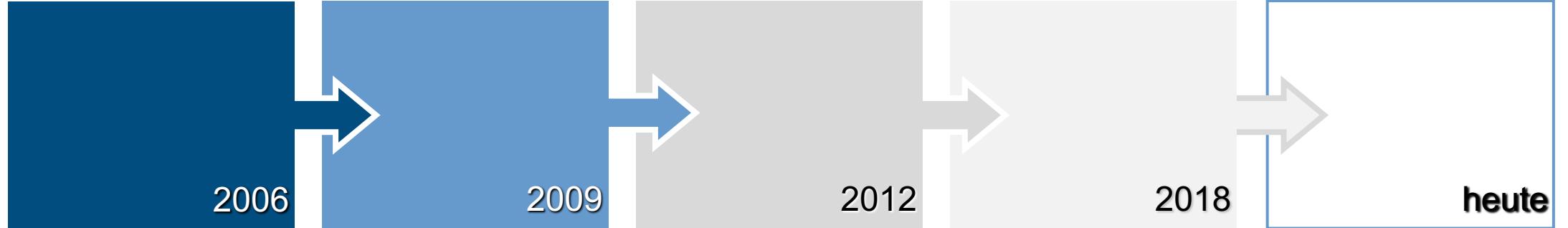
**GWT im Unternehmen**

**Zusammenfassung und Fazit**





## GWT - Historie



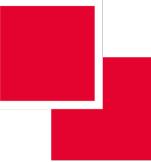
- Version 1 wurde released

- Version 2 wurde released

- Übergabe der Steuerung von GWT an das Steering Committee

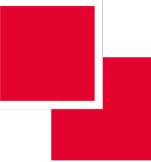
- J2CL seit November 2018 in einer Public-Beta-Phase

- Arbeiten an GWT 3 (J2CL-Basis) laufen
- GWT 2.x Support für mehrere Jahre garantiert (Vertispan)
- Google hat einen LTS für J2CL von mindestens 5 Jahren genannt



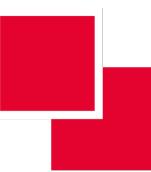
## Was ist GWT?

- GWT/J2CL ist in erster Linie ein Transpiler (Java -> JS)
- Code-Generierung (GWT: Generator / J2CL: APT)
- Debugging über SuperDevMode (SDM)
- Module für:
  - Data-Binding
  - I18N
  - Widgets
  - Activities & Places
  - DOM-Manipulation, native Event-Handling, etc.
  - Ressourcen (Images, CSS, etc.)
  - JSNI – Java Script Native Interface
  - JsInterop für die Anbindung von JS-Libraries (Nachfolger für JSNI)
  - Elemental2 – JS-API in Java
- Langlebig



## Was ist J2CL?

- Neue Implementierung des Transpilers (Java -> JS)
- Mit folgenden Änderungen:
  - Wegfall der Permutationen (es gibt nur noch eine)
  - Keine Code-Generatoren (APT verwenden)
  - Keine GWT-Properties mehr (System Properties verwenden)
  - Keinen Module Descriptor mehr
    - Keine Replacement-Rules, etc.
  - Kein JSNI mehr
- → Alle bestehenden GWT-Module sind anzupassen
  - rund 80% der Module sind bereits umgestellt
  - J2CL-Tests sind in Arbeit
  - Migration der Module in ein zentrales Repo geplant
- LTS für mindestens 5 Jahre seitens Google



## JsInterop – Ersatz für JSNI

- Ermöglicht die Interoperabilität zwischen Java und JS
- Ermöglicht die Anbindung bestehender JS-Libs
- Ermöglicht die Nutzung von Java-Klassen direkt in JavaScript

```
@JsType(isNative = true)
public class AlertSystem {

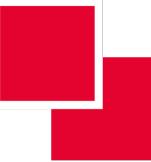
    public String message;

    public native void alert();

}

@JsType(name="WarningSystem",
        namespace = "examples")
public class AlertSystem {
    @JsProperty(name="msg")
    public String message;

    @JsMethod(name="warn")
    public void alert() {
        Window.alert(message);
    }
}
```

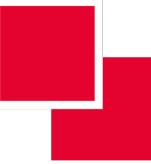


## Elemental2 JS-API mit Java programmieren

- Bildet die JavaScript-API in Java ab
- Basiert auf JsInterop

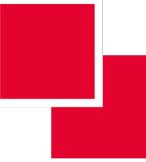
```
import static elemental2.dom.DomGlobal.fetch;
import static elemental2.dom.DomGlobal.console;
import elemental2.dom.Response;

fetch("https://randomuser.me/api/?gender=female&results=1")
    .then(Response::json)
    .then(data -> {
        console.log(Global.JSON.stringify(data));
        return null;
    })
    .catch_(error -> {
        console.log(error);
        return null;
    });
}
```



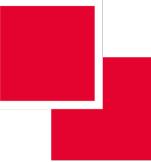
## GWT-Eco-System

- Open-Source & kommerzielle Libraries
  - Erweitern die Funktionalität von GWT
  - Binden JS-Libs an
    - Vue-GWT
    - gwt-ol (Wrapper für Open-Layers)
    - Sencha GXT 5
    - Und viele mehr ...
  - Stellen alternative Implementierungen bereit:
    - RestyGWT, auto-rest, **domino-rest**
    - MVP4G, MVP4G2, domino-mvp, **Nalu**
    - SmartGWT, Sencha GXT, Elemento, Vue-GWT, **domino-ui**
    - Dagger 2 (GIN-Nachfolger)
  - Und viele mehr ...



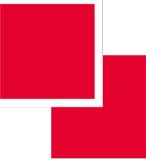
## RIA – Rich Internet Application

- Motivation:
  - Server als Daten-Factory
  - Browser als Laufzeitumgebung des Clients
  - Verhalten einer Desktop-Anwendung (PWA)
    - Bietet dem User die beste UX
- SPA – Single Page Application
- Session liegt im Browser (kein Zwang)
- Verwendung von Stateless Services
- Rendering erfolgt auf dem Client
- Applikations-Code wird beim Start geladen
- Danach werden nur noch Daten vom Server gelesen
- Die Kommunikation mit dem Server ist asynchron
- Stateless Services auf dem Server



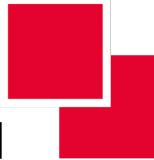
## Wie entwickelt man mit GWT zukunftsicher?

- Weiterhin auf GWT 2.8.2
- Zukunftssichere Implementierungen einsetzen
  - Verwenden der bereits umgestellten Module
  - Keine Klassen verwenden, die von GWT-Widgets erben
  - Third Party Libs auswählen, die mit J2CL und GWT laufen
  - Keine Generatoren verwenden (nur APT)
  - Nur System-Properties verwenden
- Beispiel:  
Eine Anwendung auf Basis von: **domino-rest, domino-ui & Nalu**



## domino-rest – ein REST-Client

- Verwendet das JSON-Protokoll
- Läuft mit Java, GWT 2.8.2 & J2CL  
(kann auf dem Server und auf dem Client verwendet werden!)
- Sehr einfache Implementierung
- Unterstützt Collections und Polymorphismus
  
- Repo:
  - <https://github.com/DominoKit/domino-rest>
- Kontakt zu den Devs:
  - <https://gitter.im/domino-gwt/domino-ui>



## Initialisieren & DTOs

- Initialisieren des Domino-Rest-Context
- Erzeugen der Mapper für die Pojos durch die Annotation mit

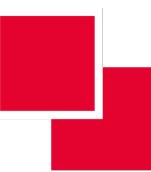
**@JSONMapper**

```
DominoRestConfig.initDefaults();
```

```
@JSONMapper
public class Movie {

    private String name;
    private int rating;
    private String bio;
    private String releaseDate;

    // setters and getters
}
```



## Service Definition

- Erzeugen einer Service Definition für jeden Rest-Service
  - Interface erzeugen
  - Mit @RequestFactory annotieren
  - Methoden definieren
  - Methoden mit @Path und der Aufruf-Methode annotieren

```
@RequestFactory
public interface MoviesService {

    @Path("library/movies/:movieName")
    @GET
    MovieResponse getMovieByName(String movieName);

    @Path("library/movies")
    @GET
    List<Movie> listMovies();

    @Path("library/movies/:name")
    @PUT
    void updateMovie(@RequestBody Movie movie);
}
```



## Verwendung des generierten Client

- Der Annotation Processor erzeugt für jedes Interface eine Factory
- In der **onSuccess**-Methode erfolgt die Verarbeitung bei einem erfolgreichen Call
- In der **onFailure**-Methode erfolgt die Verarbeitung bei einem nicht erfolgreichen Call
- Versenden erfolgt über **send()**

```
MoviesServiceFactory.INSTANCE
    .getMovieByName("hulk")
    .onSuccess(movie -> {
        //do something on success
    })
    .onFailed(failedResponse -> {
        //do something on error
    })
    .send();

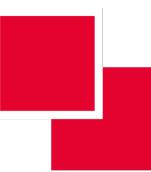
MoviesServiceFactory.INSTANCE
    .listMovies()
    .onSuccess(movies -> {
        //do something on success
    })
    .onFailed(failedResponse -> {
        //do something on error
    })
    .send();

MoviesServiceFactory.INSTANCE
    .updateMovie(movie)
    .onSuccess(aVoid -> {
        //do something on success
    })
    .onFailed(failedResponse -> {
        //do something on error
    })
    .send();
```



## domino-ui – eine Widget-Lib

- Kompatibel mit GWT 2.8.2 und J2CL
- Basiert auf Elemento
- Ist Responsive
- Keine Abhangigkeit zu einer externen JavaScript-Lib
- Feature-Rich
- Editor-Support, Fehler-Anzeigen, Tooltips, Validatoren
  
- Demo:
  - <https://dominokit.github.io/domino-ui-demo/index.html?theme=indigo#home>
  
- Repo:
  - <https://github.com/DominoKit/domino-ui>
  
- Kontakt zu den Devs:
  - <https://gitter.im/domino-gwt/domino-ui>



## Features

- Type Safe
- Rich Forms
  - Cards, Dialog, Window, Alerts, Info-Box
  - Date- und Time-Picker
  - Buttons
  - Lists
  - Loaders
  - Breadcrumb
  - Notifications
  - Tree
- Rich Data Table
- Typography & Font icons

**BASIC TABLE - FIXED**  
Fixed tables will use the specified column width and will have scrolls when elements exceeds the body height.

#	Status	First name	Gender	Eye color	Balance	Email
8	✗	Brianna Lyons	♂	green	<div style="width: 80%; background-color: green;"></div>	briannalyo...
9	✓	Serrano Green	♂	green	<div style="width: 60%; background-color: orange;"></div>	serranogre...
10	✗	Matilda Dejesus	♀	blue	<div style="width: 90%; background-color: green;"></div>	matildadej...
11	✗	Camacho Solis	♂	blue	<div style="width: 50%; background-color: blue;"></div>	camachos...
12	✗	Britney Moss	♂	blue	<div style="width: 40%; background-color: orange;"></div>	britneymos...
13	✓	Berger Odom	♀	blue	<div style="width: 70%; background-color: orange;"></div>	bergerod...
14	✗	Dionne Hogan	♂	blue	<div style="width: 30%; background-color: orange;"></div>	dionnehog...
15	✗	Booth Whitaker	♂	green	<div style="width: 95%; background-color: green;"></div>	boothwhita...

**SUGGEST BOX**

Local suggest store

Your friend name  **+ ADD FRIEND**

Add friend name as suggestion

Your friends

Type any letter and see suggestions

Dynamic suggest store

Country

**Shipment Details \***

Transshipment  Permitted

Latest Date Of Shipment  dd-MM-yyyy

Not permitted  Permitted

Partial Shipments  Permitted

Shipment By

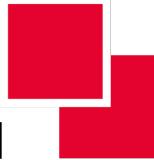
Not permitted  Permitted

Shipment From

Shipment To

Terms Of Delivery (Incoterms 2010)

Place Of Destination



## Domino-UI Beispiele

**DominoDo**

**Todo**

- All Tasks
- Today's tasks
- Next week

**Important**

**Normal**

**GWT**

**Domino UI**

**Nalu project**

**Movies**

**Resolved**

**All Tasks**

**Text box icons** Domino UI 2019 June 26, Wednesday

TextBox setRightAddon, removeRightAddon and setRightAddon again throws Exception

Bug Milestone 1.0

**ATTACHMENTS** 2 Files

**Router/Permissions** Nalu MVP 2019 June 28, Friday

Router: create the possibility to work with permissions in case of routing

Enhancements Router Milestone 1.0

**ATTACHMENTS** 3 Files

**IE 11** Domino UI

+ ADD TASK

**Nalu Initializer** 1.3.0.0007 Powered by

**Project Meta Data**

Group ID	com.example	Artifact ID	MyTestProject
GWT Version	2.8.2	Widget Set	use Domino UI widgets
Maven	use a Single Maven Module		

**Application Meta Data**

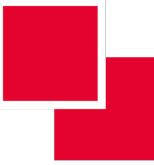
Generate Application Loader class	Generate Debug support (in development mode)
Generate Login screen and Login filter	Generate Error Screen
Use hash in URL	

**Screen Meta Data**

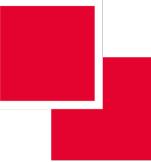
Component name	Route	Start screen	Confirmation
Screen01	screen01	No <input checked="" type="checkbox"/> Yes <input type="checkbox"/>	No <input checked="" type="checkbox"/> Yes <input type="checkbox"/>
Screen02	screen02	No <input checked="" type="checkbox"/> Yes <input type="checkbox"/>	No <input checked="" type="checkbox"/> Yes <input type="checkbox"/>
Screen03	screen03	No <input checked="" type="checkbox"/> Yes <input type="checkbox"/>	No <input checked="" type="checkbox"/> Yes <input type="checkbox"/>
Screen04	screen04	No <input checked="" type="checkbox"/> Yes <input type="checkbox"/>	No <input checked="" type="checkbox"/> Yes <input type="checkbox"/>
Screen05	screen05	No <input checked="" type="checkbox"/> Yes <input type="checkbox"/>	No <input checked="" type="checkbox"/> Yes <input type="checkbox"/>

**GENERATE**

Nalu@Github Generator Documentation Issues OPEN CHAT

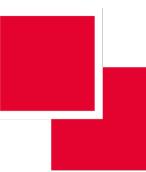


# DOMINO-UI DEMO



## Nalu Routing based App Framework

- Kompatibel mit GWT 2.8.2 und J2CL
  - Routing based Navigation
  - Unterstützt Vor-, Zurück- und Reload-Button
  - Filter-Support
  - Multi Shell Support
  - Separation of Views in Controller & Component
  - Controller Life-Cycle
  - Controller Caching
  - Loader-Support
  - Handler-Support
  - PopUp-Support
  - Debug-Support (DevMode)
  - Tracking
  - Eventbus, Context & Router
  - Application Loader
  - Unterstützt jede Art von Widget-Libs
  - Keine Einschränkung bei der Wahl der Server-Kommunikation
- 
- Repo:
    - <https://github.com/nalukit/nalu>
  - Kontakt zu den Devs:
    - <https://gitter.im/Nalukit42/Lobby>



## Der Einstieg: NaluInitializer

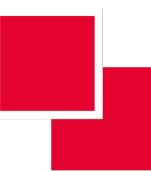
- Erzeugt ein Nalu-Basis Projekt
- Single- oder Multi-Maven Project
- Aktuell: Unterstützung für vier Widget-Libs
- Beliebige Anzahl an Screens
- Generierung einer Login-Komponente
- Generierung einer Error-Komponente

The screenshot shows the NaluInitializer web application interface. At the top, it displays "NaluInitializer 1.3.0.0007" and "Powered by". The main area is divided into three sections: "Project Meta Data", "Application Meta Data", and "Screen Meta Data".

- Project Meta Data:** Includes fields for Group ID (com.example), Artifact ID (MyTestProject), GWT Version (2.8.2), Maven (use a Single Maven Module), and Widget Set (use Domino-UI widgets).
- Application Meta Data:** Contains checkboxes for generating Application Loader class, Debug support, Login screen, Error Screen, and hash in URL.
- Screen Meta Data:** A table listing five screens with columns for Component name, Route, Start screen, and Confirmation. Each row has a "Yes" or "No" switch for each column.

At the bottom, there is a "GENERATE" button and links for "Nalu@Github", "Generator Documentation", "Issues", and "OPEN CHAT".

■ <http://www.mvp4g.org/gwt-boot-starter-nalu/GwtBootStarterNalu.html>



## ... das generierte Projekt

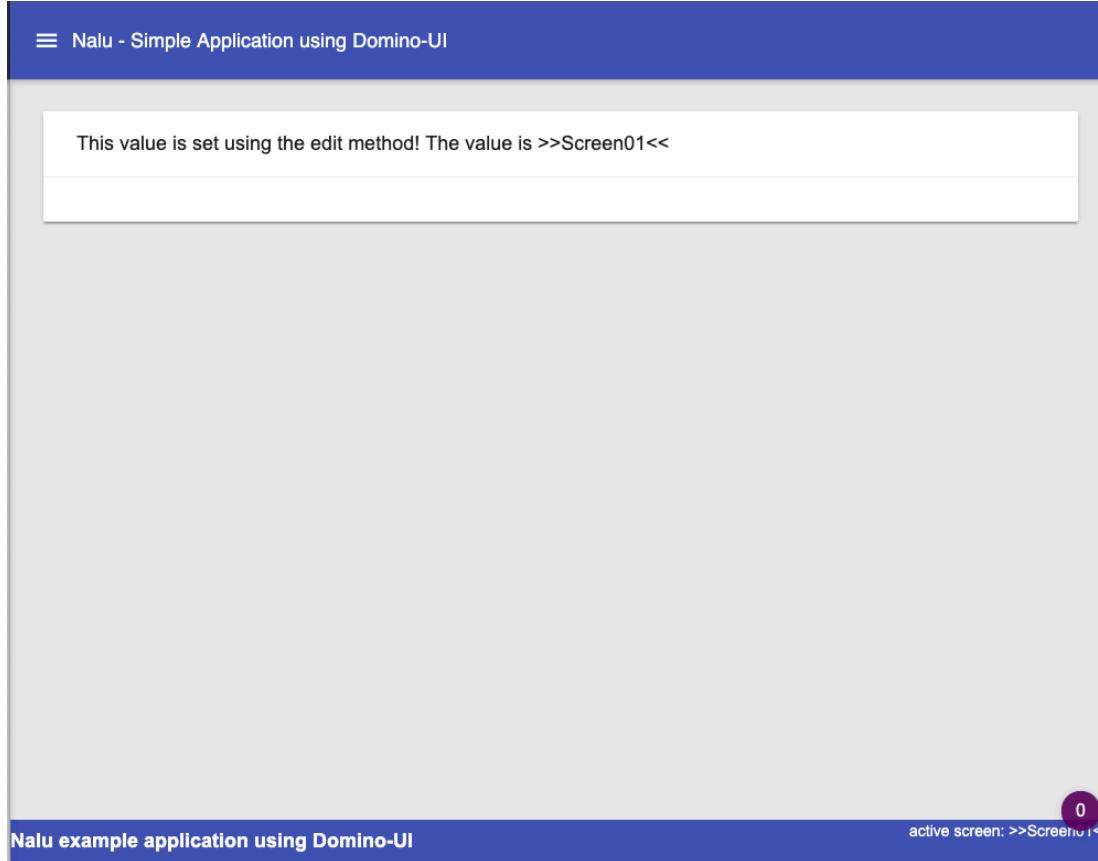
≡ Nalu - Simple Application using Domino-UI

This value is set using the edit method! The value is >>Screen01<<

Nalu example application using Domino-UI

active screen: >>Screen01<<

0



≡ Nalu - Simple Application using Domino-UI

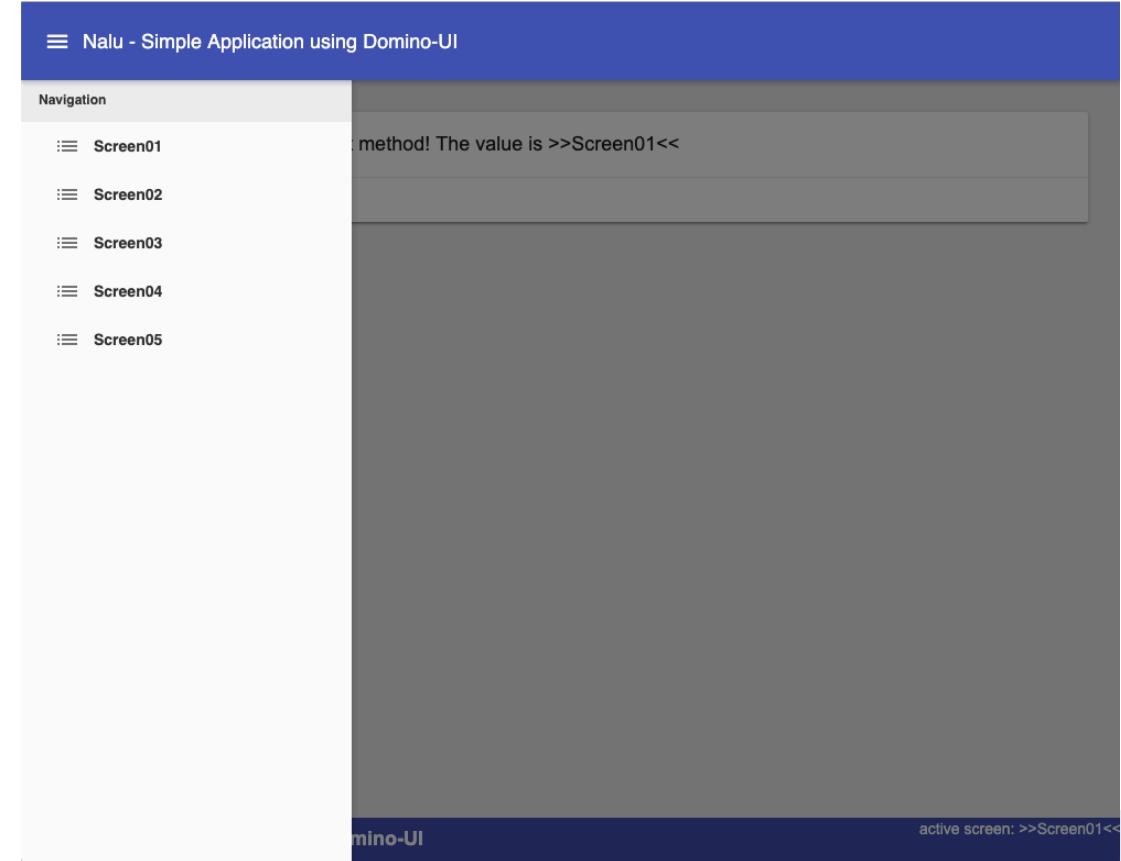
Navigation

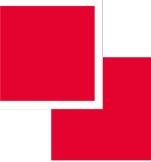
- ≡ Screen01
- ≡ Screen02
- ≡ Screen03
- ≡ Screen04
- ≡ Screen05

This value is set using the edit method! The value is >>Screen01<<

Nalu example application using Domino-UI

active screen: >>Screen01<<





# EIN BLICK IN EIN GWT-PROJEKT

Demo: <https://github.com/NaluKit/nalu-examples/tree/master/devkexample>



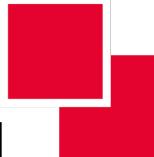
## Elemento

- Type safe builders, HTML templates and other goodies for GWT Elemental2
- Repo:
  - <https://github.com/hal/elemento>

```
<section class="main">
  <input class="toggle-all" type="checkbox">
  <label for="toggle-all">Mark all as complete</label>
  <ul class="todo-list">
    <li>
      <div class="view">
        <input class="toggle" type="checkbox" checked>
        <label>Taste Elemento</label>
        <button class="destroy"></button>
      </div>
      <input class="edit">
    </li>
  </ul>
</section>
```

```
import static org.jboss.gwt.elemento.core.Elements.*;
import static org.jboss.gwt.elemento.core.InputType.checkbox;
import static org.jboss.gwt.elemento.core.InputType.text;

HTMLElement section = section().css("main")
  .add(input(checkbox).id("toggle-all").css("toggle-all"))
  .add(label()
    .apply(l -> l.htmlFor = "toggle-all")
    .textContent("Mark all as complete"))
  .add(ul().css("todo-list"))
    .add(li()
      .add(div().css("view"))
        .add(input(checkbox)
          .css("toggle")
          .apply(cb -> cb.checked = true))
        .add(label().textContent("Taste Elemento")))
      .add(button().css("destroy")))
    .add(input(text).css("edit")))
  .get();
```



## Elemento Support für HTML-Templates

### Template

```
<!doctype html>
<html lang="en">
<head>
  <link rel="stylesheet" href="/node_modules/todomvc-common/base.css">
  <link rel="stylesheet" href="/node_modules/todomvc-app-css/index.css">
</head>
<body>
<section data-element="todos" class="todoapp">
  <header class="header">
    <h1>todos</h1>
    <input data-element="newTodo" class="new-todo" placeholder="What needs to be done?" autofocus>
  </header>
  <section data-element="main" class="main">
    <input data-element="toggleAll" class="toggle-all" id="toggle-all" type="checkbox">
    <label for="toggle-all">Mark all as complete</label>
    <ul data-element="list" class="todo-list">
      <!-- Todo items are mapped to an extra template class -->
    </ul>
  </section>
  <footer data-element="footer" class="footer">
    <span data-element="count" class="todo-count"><strong>0</strong> item left</span>
    <ul class="filters">
      <li><a data-element="all" href="#">All</a></li>
      <li><a data-element="active" href="#/active">Active</a></li>
      <li><a data-element="completed" href="#/completed">Completed</a></li>
    </ul>
    <button data-element="clearCompleted" class="clear-completed">Clear completed</button>
  </footer>
</section>
[...]
</body>
</html>
```

### Template Class

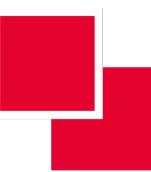
```
@Templated("Todo.html#todos")
abstract class ApplicationElement implements IsElement<HTMLElement> {

  static ApplicationElement create() {
    return new Templated_ApplicationElement();
  }
}
```

### Data-Binding

```
@Templated("Todo.html#todos")
abstract class ApplicationElement implements IsElement<HTMLElement> {

  @DataElement HTMLElement newTodo;
  @DataElement HTMLElement main;
  @DataElement HTMLElement toggleAll;
  @DataElement HTMLElement list;
  @DataElement HTMLElement footer;
  @DataElement HTMLElement count;
  @DataElement("all") HTMLElement filterAll;
  @DataElement("active") HTMLElement filterActive;
  @DataElement("completed") HTMLElement filterCompleted;
  @DataElement HTMLElement clearCompleted;
}
```



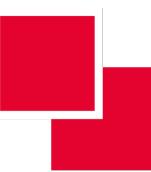
## Vue-GWT

- Vue.js Components/Custom Elements in Java with GWT
- Erzeugt Input für vue.js
- Repo:
  - <https://github.com/VueGWT/vue-gwt>
- Kontakt zu den Devs:
  - <https://gitter.im/domino-gwt/domino-ui>

### Component

```
@Component
public class SimpleLinkComponent implements IsVueComponent {
    @Data String linkName = "Hello Vue GWT!";
}
```

- Erzeugen einer Klasse
- Annotieren der Klasse mit **@Component**
- Mittels **@Data** können Properties sichtbar gemacht



## Vue-GWT

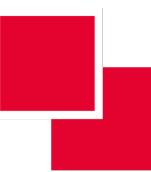
### HTML-Template

```
<a href="https://github.com/VueGWT/vue-gwt">  
    {{ linkName }}  
</a>
```

- Das Template muss denselben Namen wie die Java-Klasse besitzen
- Muss sich im selben Package befinden wie die Java-Klasse
- Zur Laufzeit wird der String in den doppelten geschweiften Klammer als Name des Properties verwendet

### Verwendung

```
public class RootGwtApp implements EntryPoint {  
    public void onModuleLoad() {  
        VueGWT.init();  
        SimpleLinkComponent simpleLinkComponent = Vue.attach("#simpleLinkComponentContainer",  
            SimpleLinkComponentFactory.get());  
    }  
}
```



## Vue-GWT Binding & Conditional

### Binding Element Attributes

```
@Component
public class LinkComponent implements IsVueComponent {
    @Data String linkName = "Hello Vue GWT!";
    @Data String linkTarget = "https://github.com/VueGWT/vue-gwt";
}
```

```
<a v-bind:href="linkTarget">
    {{ linkName }}
</a>
```

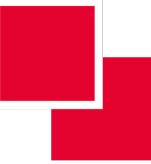
### Conditional

```
<div v-if="visible">
    The property "visible" is set to true!
</div>
```

```
@Component
public class CanHideComponent implements IsVueComponent {
    @Data boolean visible = true;
}
```

### Ansprechen in GWT

```
canHideComponent.visible = false;
```



## Vue-GWT Loops

### Pojo

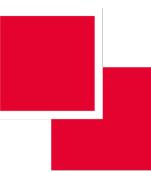
```
public class Todo {  
    private String text;  
  
    public Todo(String s) {  
        this.text = s;  
    }  
  
    public String getText() {  
        return text;  
    }  
}
```

### Component

```
@Component  
public class SimpleTodoListComponent implements IsVueComponent, HasCreated {  
    @Data  
    @JsProperty  
    List<Todo> todos = new LinkedList<>();  
  
    @Override  
    public void created() {  
        this.todos.add(new Todo("Learn Java"));  
        this.todos.add(new Todo("Learn Vue GWT"));  
        this.todos.add(new Todo("Build something awesome"));  
    }  
}
```

### Template

```
<vue-gwt:import class="com.mypackage.Todo"/>  
<ol>  
    <li v-for="Todo todo in todos">  
        {{ todo.getText() }}  
    </li>  
</ol>
```



## Vue-GWT Handling User Input

### Template

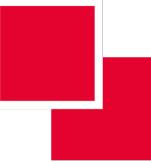
```
<div>
  <p>{{ message }}</p>
  <button v-on:click="addExclamationMark()">Add an explanation mark</button>
</div>
```

### Component

```
@Component
public class ExclamationComponent implements IsVueComponent {
    @Data String message = "Hello Vue GWT!";

    @JsMethod // Notice the @JsMethod annotation to expose this method to our template
    public void addExclamationMark() {
        this.message += "!";
    }
}
```

Mittels **exclamationComponent.addExclamationMark()** kann die Java Methode auf der Browser-Console aufgerufen werden



## Vue-GWT V-Model Directive

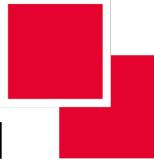
### Template

```
<div>
  <p>{{ message }}</p>
  <input type="text" v-model="message">
</div>
```

### Component

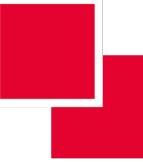
```
@Component
public class MessageComponent implements IsVueComponent {
    @Data String message = "Hello Vue GWT!";
}
```

Two-Data-Binding zwischen einem Textfeld und den Applicaiton Field



## Überraschung ...





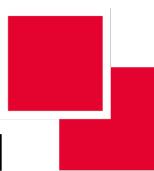
## Was war passiert ...

- Entwickler von **event-stream** hat die Weiterentwicklung an einen anderen Entwickler übertragen
- Dieser Entwickler hat der Komponente Schad-Code "untergejubelt"
- Das Paket **event-stream** wird über **npm** in der Woche rund 2 Million mal geladen
  
- Eine Recherche zeigt: Verteilung von Schadcode über npm kein Einzelfall
- Ein weiteres Beispiel:
  - Mitte 2018 wurde zwei ESLint-Packages kompromittiert
  - Angriff auf npm Tokens

<https://www.golem.de/news/event-stream-populaeres-npm-paket-verteilt-schadsoftware-1811-137935.html>

==> Herausforderung für die Security-Abteilung!

Beispiel: **create-react-app** besitzt rund 1770 indirekte Abhängigkeiten



## Wie sieht es mit dieser Problematik bei GWT aus?

### Community

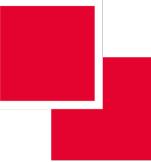
- **Open Source Nutzung aber Closed Source Entwicklung**
  - Nur ein kleiner ausgewählter Kreis an Entwicklern dürfen den Code verändern.
  - Standardisierter Code-Reviewprozess (Pull-Request-Verfahren)

### 3rd Party Libs

- Geringe Anzahl an Abhängigkeiten zu weiteren Bibliotheken erleichtert den Security-Review Prozess.
- GWT Code hat keine Abhängigkeiten zu 3rd Party JavaScript-Bibliotheken
- Mit GWT 3.0 Transpiler (J2CL) und dem ClosureCompiler wird die Einbindung weiter optimiert => Es wird nur noch der erforderliche Code aus den Bibliotheken geladen

### Security

- Keine bekannten Security Issues
- Alte Jetty-Version ist nur für die lokale Entwicklung und nicht für den produktiven Einsatz vorgesehen!
- Security Issues können durch Entwickler implementiert werden - wie bei anderen Frameworks auch



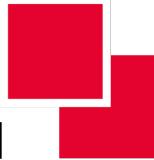
## Bewertung von GWT

### PRO

- ✓ Selbe Technologie auf Server und Clientseite (Datenmodellierung reusen etc.)
- ✓ Full-Stack Framework
- ✓ Nähe zur Programmiersprache Java
- ✓ Nur Basiskenntnisse von CSS und HTML erforderlich
- ✓ Einsatz bekannter IDEs / Java-Toolings
- ✓ Erhöhte Wartbarkeit durch strukturierten Code
- ✓ Gute Standardisierungsmöglichkeiten
- ✓ Template Engine ermöglicht die Zusammenarbeit mit klassischen Webentwicklern
- ✓ Unterstützung aller gängigen Browserversionen
- ✓ Unterstützung von „Lazy-Loading“

### CONTRA

- Eher schwergewichtig und somit nicht für alle Anwendungsfälle optimal
- Gewisser „Overhead“ bei kleinen Anwendungen
- Compilezeit der Artefakte zur Zeit relativ hoch
- „Klassische“ Webentwickler tun sich mit der Entwicklungsart schwer



## Gegenüberstellung

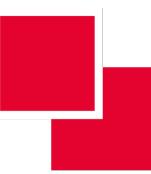
### GWT

- Rich Internet Application (RIA)
- Full-Stack Framework
- Beliebige Verwendung von Widget Libraries, unabhängig von der Technologie
- Relativ Schwer gewichtet
- Nähe zur Programmiersprache Java
- Identische Programmiersprache auf Server und Client
- Wiederverwendung des Datenmodells
- Etabliertes Steering Committee
- Gut geeignet für Anwendungen mit einer langen Lebensdauer

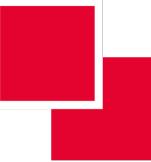
vs

### Vue.js

- Single Page Application (SPA)
- Library, kein vollständiges Framework
- Ohne Fremdbibliotheken eine reine Präsentationsschicht
- Leichtgewichtet
- Eingängig für JavaScript Entwickler
- Community etabliert sich immer mehr
- Gut geeignet für kleinere Projekte / Anwendungen mit relativ kurzer Lebensdauer

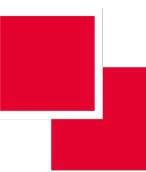


## Was bedeutet das für einen Einsatz im Unternehmen?



## Wann bietet sich der Einsatz von GWT an?

- 1 Java Skill ist primär in der Organisation vertreten.**
- 2 Man möchte den Aufbau der Anwendungsarchitektur standardisieren, um besser skalieren zu können.**
- 3 Die Anwendung weist eine hohe Komplexität auf.**
- 4 Es wird eine lange Lebensdauer der Anwendung erwartet.**
- 5 Lastverteilung von Business Logik auf Client zur Reduzierung von Server-Aufrufen.**



## Nutzen für das Unternehmen

### Skill

- Nähe zu Java reduziert die Skillanforderung an den Mitarbeiter.
- Standardisierung durch eigene Frameworks erleichtert die Nutzung von GWT.

### Wartung

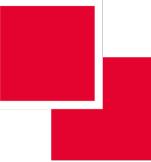
- Wartbarkeit und Erweiterungen der Anwendung
  - Analoger Aufbau interner Anwendungen durch Standardisierungsmöglichkeiten und strukturierter Code.
  - Etablierte Tools wie Find-Bugs, SolarLint und weitere Tools helfen bei der Erreichung hoher Code-Qualität

### Effizienz

- Fokussierung auf die Umsetzung fachlicher Aufgaben, nicht die Implementierung der Technik
  - Einsatz bestehender Libraries (z.B. Widget Library etc.)
  - Eigenimplementierungen nach Kosten-Nutzen Abwägungen

### Kosten

- Aspekte aus Effizienzen und Wartung wirken sich positiv auf die Gesamtkosten aus
- Langlebigkeit von GWT bildet eine Investitionssicherheit



## Zusammenfassung & Fazit

**GWT ist „älter“ (2006) aber nicht auf Eis gelegt: GWT befindet sich in einer aktiven Entwicklung**

**Standardisierungsmöglichkeiten machen es für Unternehmen attraktiv**

**„Mash-Ups“ aus GWT-Code und JS-Code heute ohne Probleme möglich**

**Die Verwendung von vorhandenen Bibliotheken ermöglichen einen Speed-Up der Entwicklung und eine Reduzierung der Kosten**

**„Best fit“-Ansatz: Wähle die richtige Technologie für die richtige Anforderungen, passend zum Unternehmen.**

Summary