

Introduction

Vaadin platform

Agenda

- What is Vaadin
- Java
- Components
- Exercise
- Tools
- Progressive Web Application

What is Vaadin?

Vaadin

Finnish word

Means female **reindeer**



What is Vaadin?

Vaadin is a web framework for **Java**. It comes with a suite of **components** and **tools** for building **Progressive Web Apps** your users will ❤️

How is Vaadin different?

The only framework focused on building **Progressive Web Apps** on the **Java** platform

Typesafe end-to-end integration gives unparalleled developer experience.

Vendor-backed open source is a **stable** foundation to build on.

Best-in-class **component** set makes building highly usable applications easy.

Java

vaadin> Platform ▾ Learn ▾ Services ▾ Community ▾ Directory Pricing Login Sign up

Start a new project

Latest Release Latest LTS Previous LTS Pre Release

Vaadin 13 Vaadin 10 Framework 8 Vaadin 14 Pre-release

Vaadin 13 is the newest stable version with the latest and greatest features. It is maintained until 1st of July 2019.

For Java developers

 Full Stack App with Spring

 Business App starter Java

 Simple App HTML & Java

 Simple App Java

 Project Base

 Add-on Component

Project Structure

src/main/java for Java sources

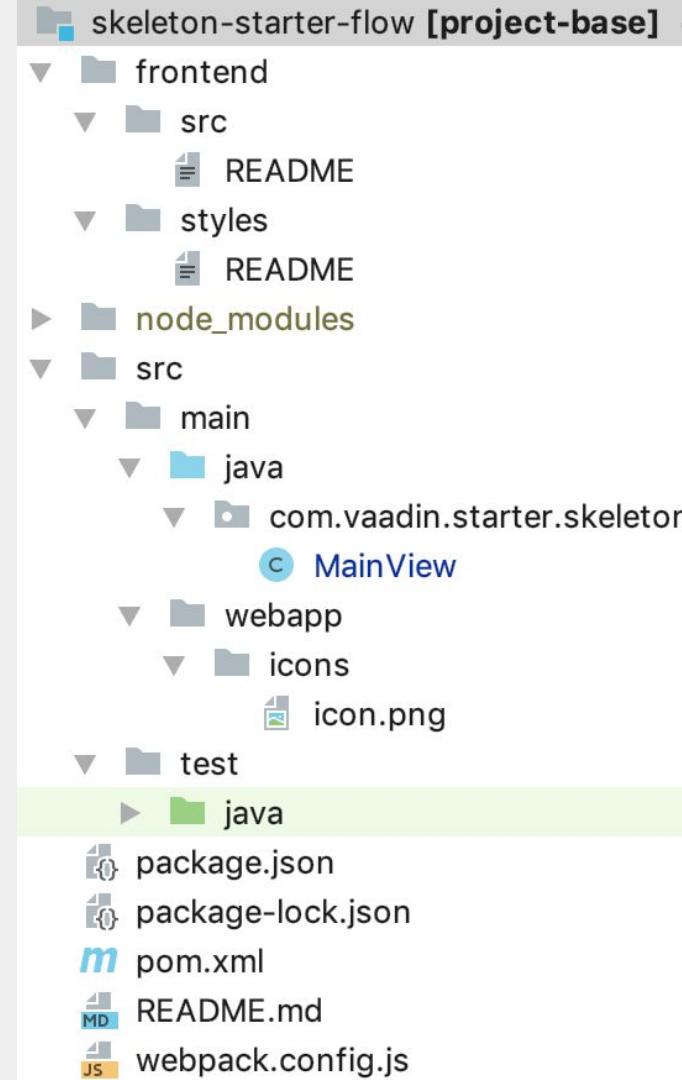
src/test/java for test sources

pom.xml Maven project configuration

frontend for client side resources, e.g.
js, HTML, CSS files

node_modules, package.json,
package-lock.json for npm

webpack.config.js auto generated
file for webpack



Hello World!

```
@Route("")
public class MainView extends VerticalLayout {
    public MainView() {
        Button button = new Button("Click me",
            e -> Notification.show("Hello World!"));
        add(button);
    }
}
```

Hello World!

Integrate with backend service

```
@Route("")
public class MainView extends VerticalLayout {
    private MessageBean messageBean = new MessageBean();

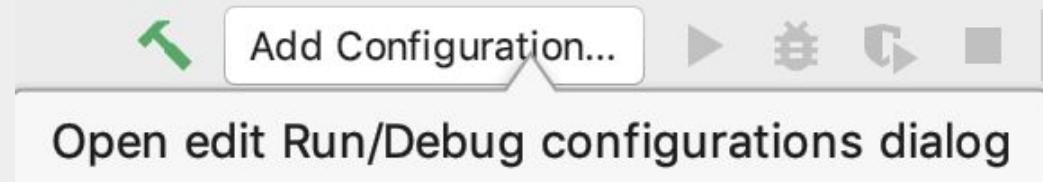
    public MainView() {
        Button button = new Button("Click me",
            e -> Notification.show(messageBean.getMessage()));
        add(button);
    }
}
```

How to run the application?

1. Run `mvn jetty:run`
2. Wait for the application to start
3. Open <http://localhost:8080> to view the application

Debug/Run (IntelliJ)

Create a new Run/Debug configuration

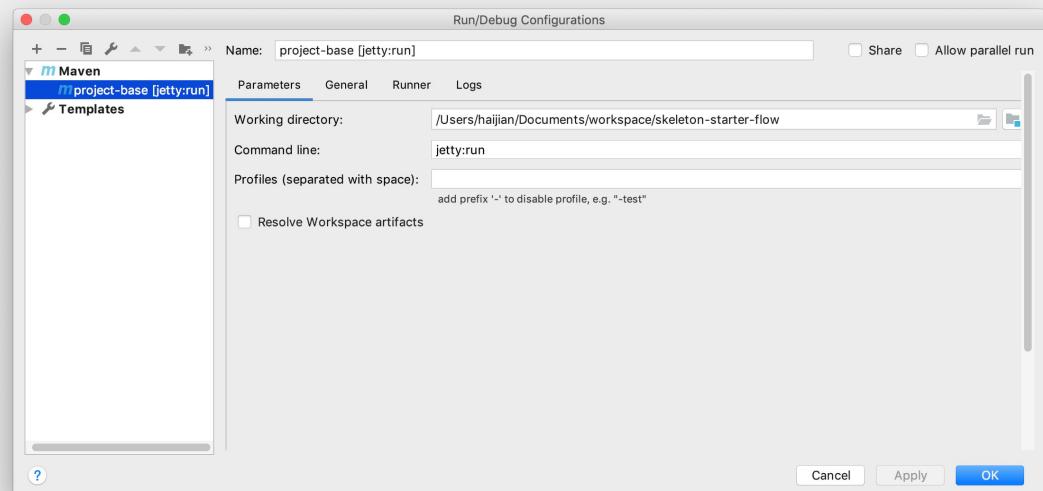


Debug/Run (IntelliJ)

Add a new Maven configuration

Specify jetty:run as the command line

Click OK



Debug/Run (IntelliJ)

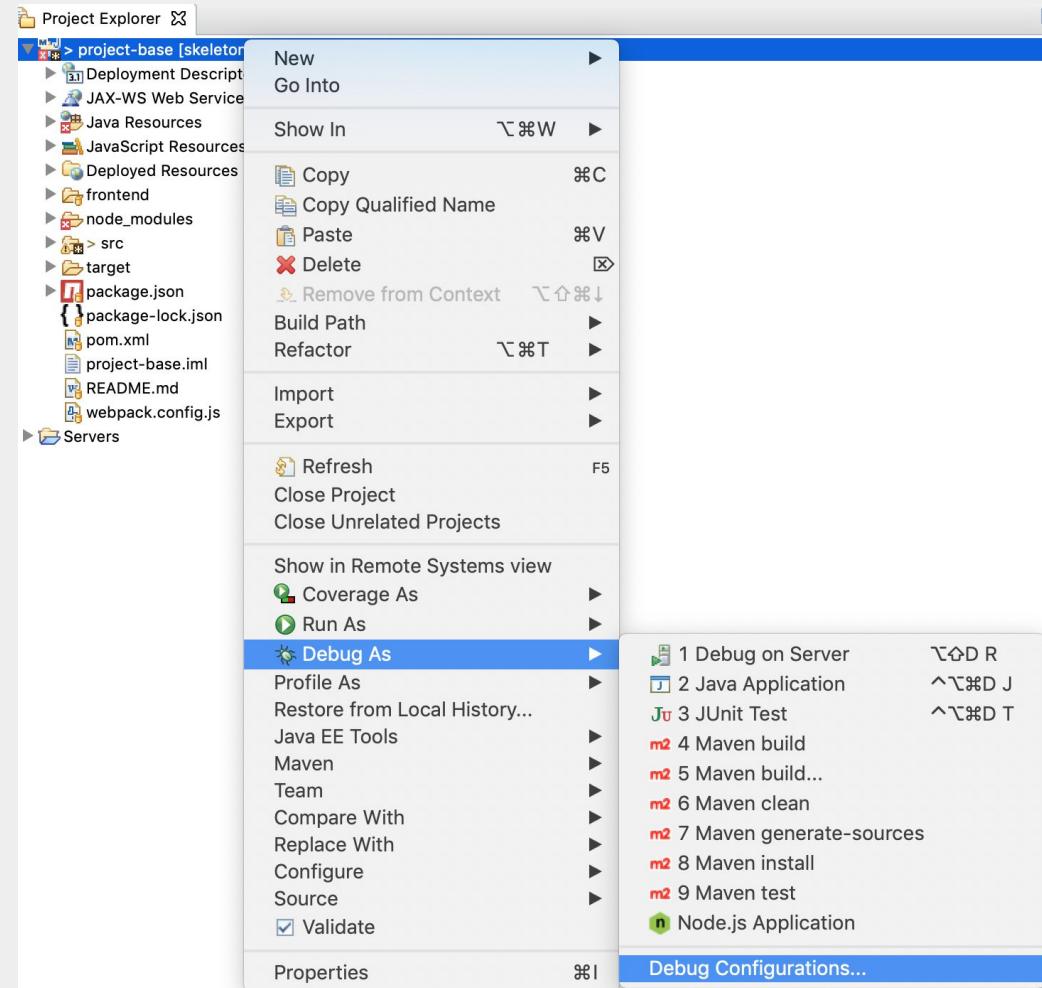
Click the debug/run button in the toolbar



Debug/Run (Eclipse)

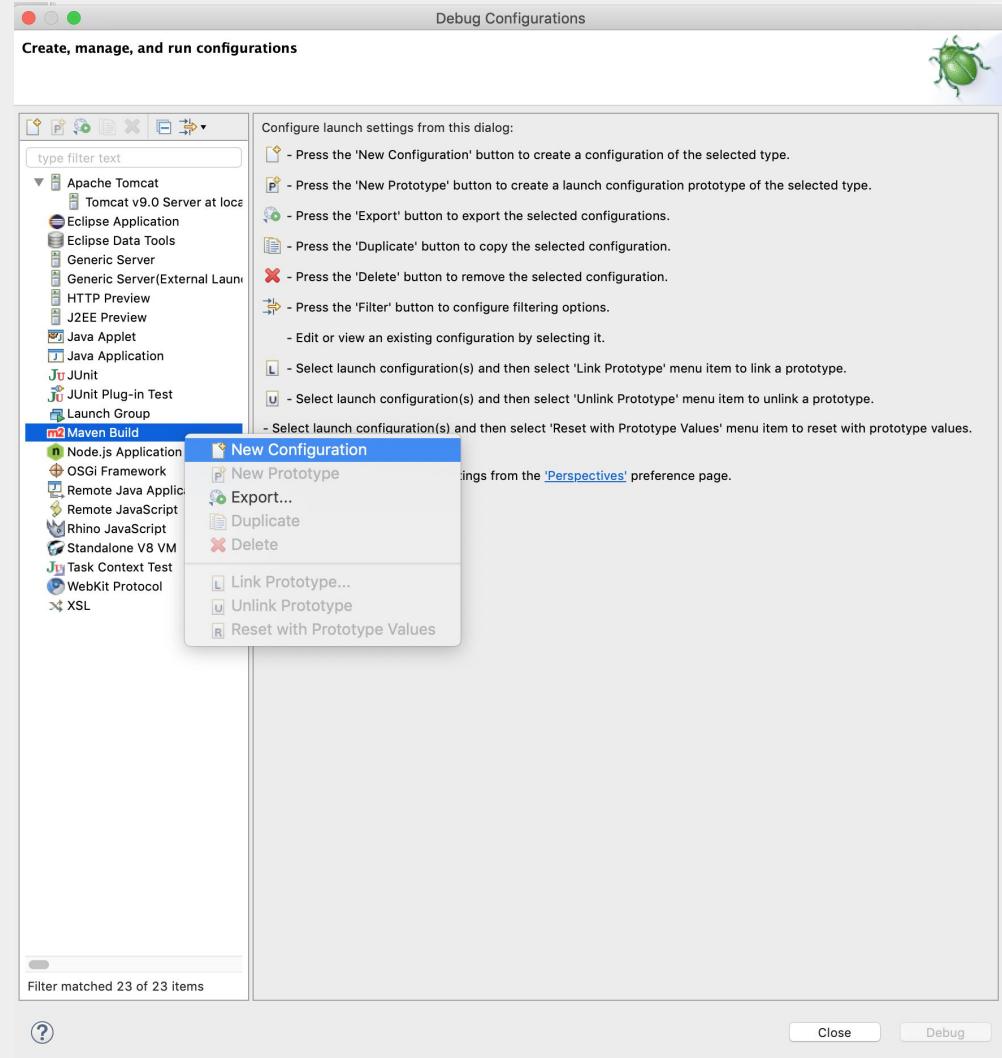
Right click on the project

Debug As -> Debug Configuration...



Debug/Run (Eclipse)

In the popup window, right click
Maven Build -> New Configuration



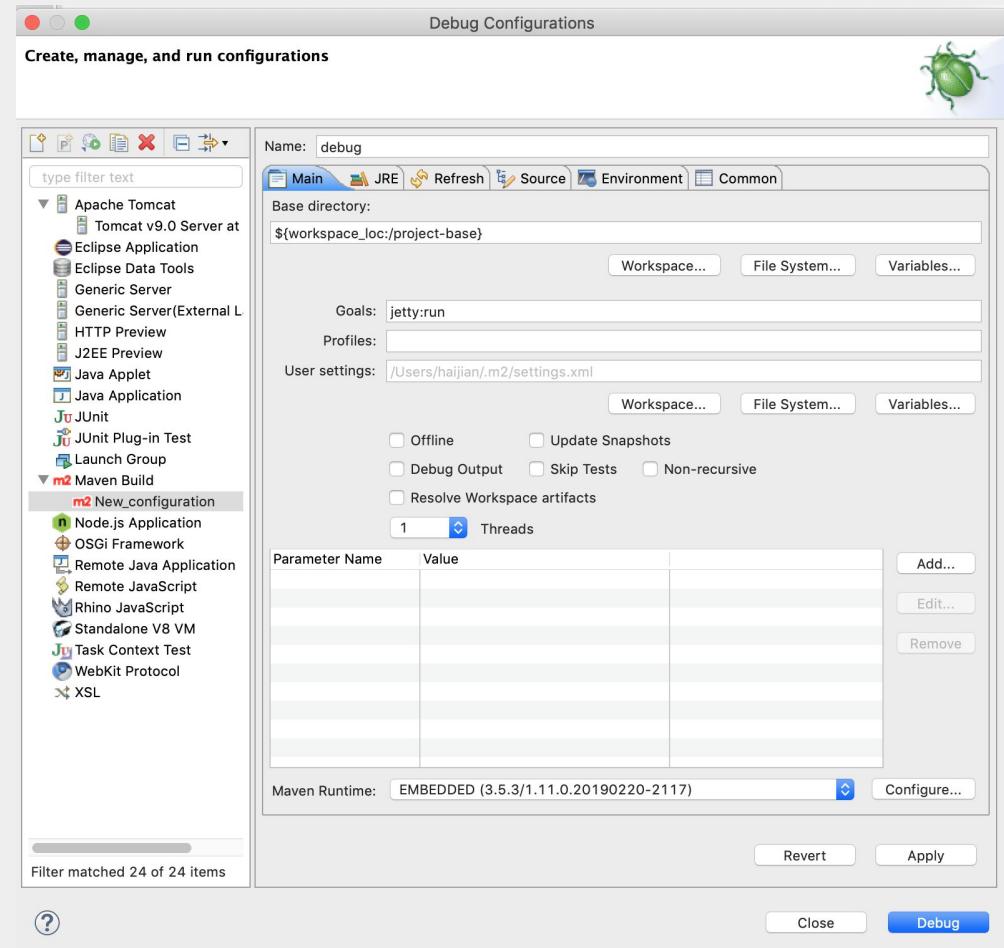
Debug/Run (Eclipse)

Give it a name

Set the Base directory as the project directory

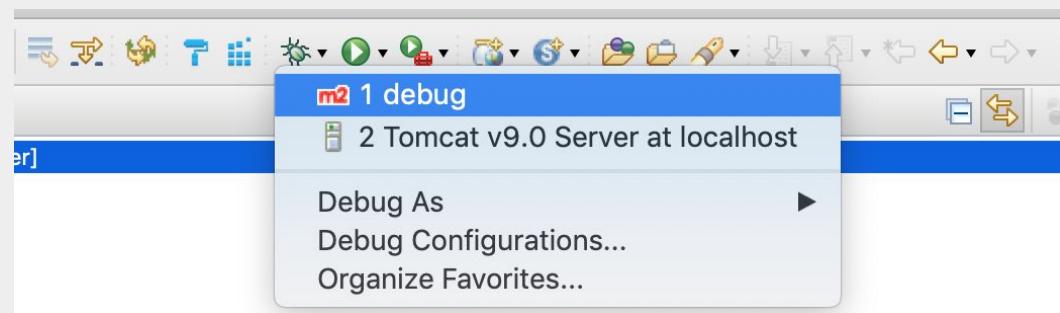
Set the goal as jetty:run

Click Apply



Debug/Run (Eclipse)

Click the debug button in the toolbar



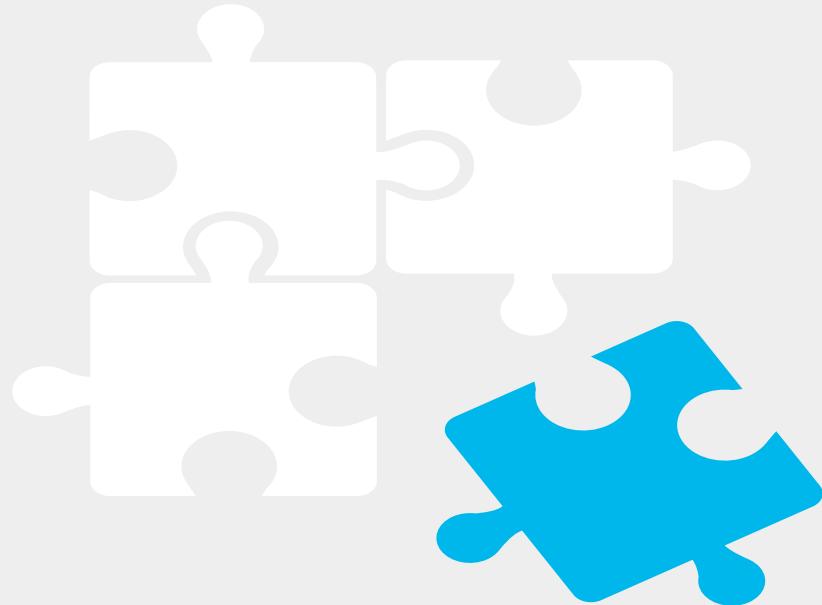
Components

Components

Built-in Components

Architecture

Web Components



Grid

	Email	Name
<input type="checkbox"/>	 kaycee.thompson@price.name	Louisa Brady
<input checked="" type="checkbox"/>	 steuber_nathanael@hotmail.com	Luke Waters
<input type="checkbox"/>	 sipes_connor@gmail.com	Francisco Saunders
<input checked="" type="checkbox"/>	 cullen_lang@elinor.io	Albert Walsh
<input checked="" type="checkbox"/>	 jakubowski_schuyler@yahoo.com	Jeff Bryant
<input type="checkbox"/>	 schmeler.delbert@maci.me	Derrick Stevenson
<input type="checkbox"/>	 lehner.raina@hotmail.com	Hester Townsend
<input type="checkbox"/>	 maeve.powlowski@louie.info	Seth Fitzgerald
<input type="checkbox"/>	 braulio.thiel@yahoo.com	Jack Walsh
<input type="checkbox"/>	 carmel_ward@mayert.com	Mildred Morgan
<input type="checkbox"/>	 buster.maggio@hotmail.com	Francis Hoffman

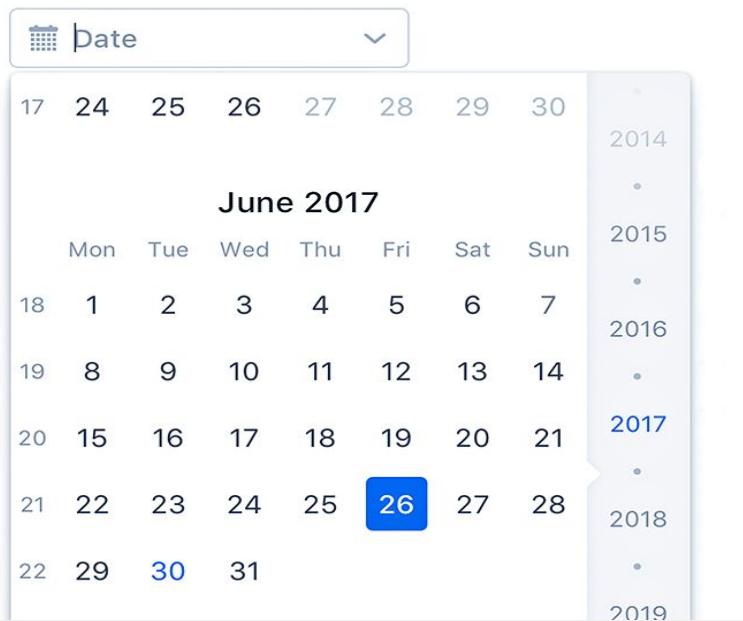
Combo box

List item

- List item
- List item
- List item
- List item

Combobox

Due •



Date Picker

Select files...

✓ test-document.doc

IMG_5272.JPG

19.7 MB: 60% (remaining time: 00:12:34)

⚠ website-mockup.pdf

An error occurred

Upload

First Name
Henry
Liam
Justin

Grid Pro



Charts

Form Inputs

Checkbox

Combo Box

Custom Field

Date Picker Email
Field

List Box

Number Field

Password Field

Radio Button

Select

Text Field

Time Picker

Upload

Visualization & Interaction

Accordion Button

Dialog

Notification

Context Menu

Grid

Progress Bar

Details

Icons

Tabs

Item

TreeGrid

Layouts

Horizontal Layout

Vertical Layout

Ordered Layout

Form Layout

Split Layout

App Layout

Login

Pro Components

Board

Cookie Consent

CRUD

Grid Pro

Charts

Rich Text Editor Spreadsheet

Confirm Dialog

HTML Components

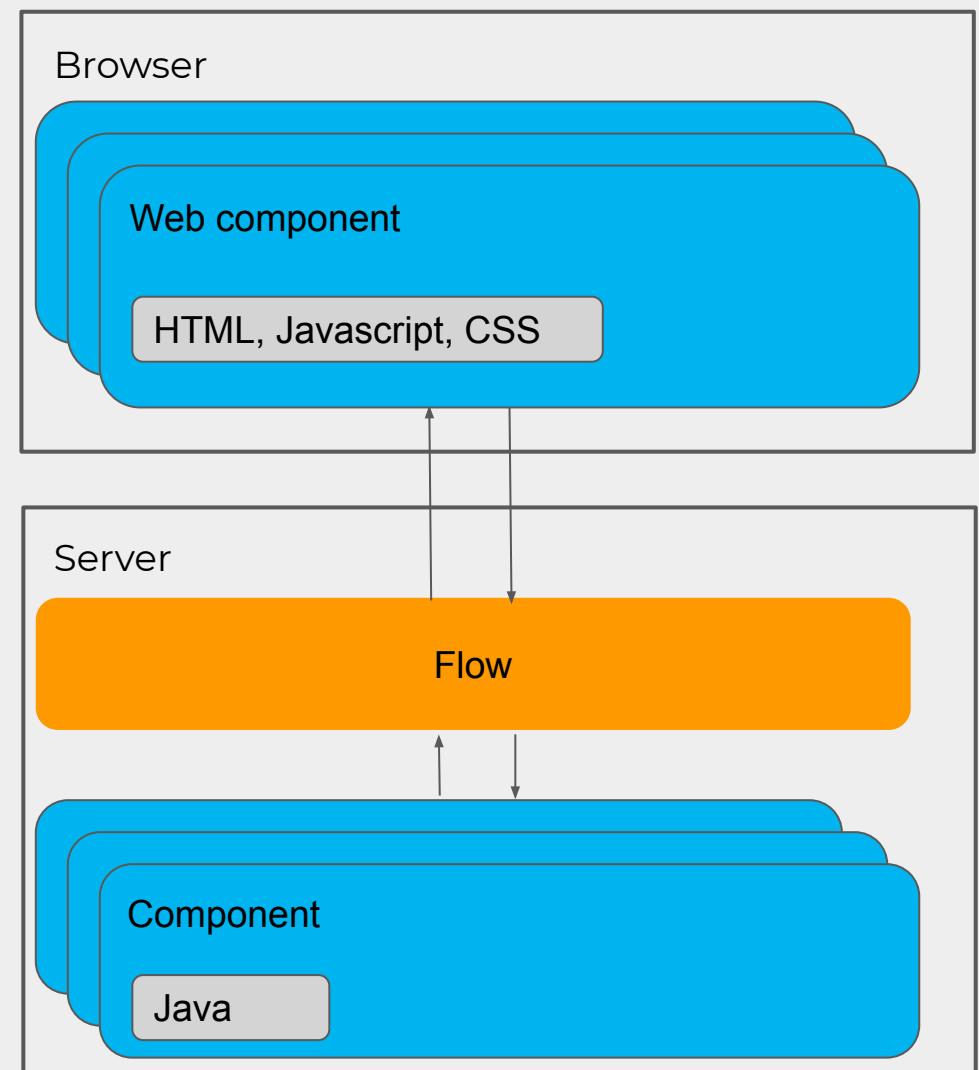
Anchor	H1	Header	NativeButton
Article	H2	Footer	Nav
Aside	H3	IFrame	OrderedList
Description	H4	Input	Paragraph
List	H5	Label	Section
Div	H6	ListItem	Span
Emphasis	Hr	Main	UnorderedList

Architecture

Client side: Web Components

Server side: Java

Communication: Flow



Web Components

Based on web standards

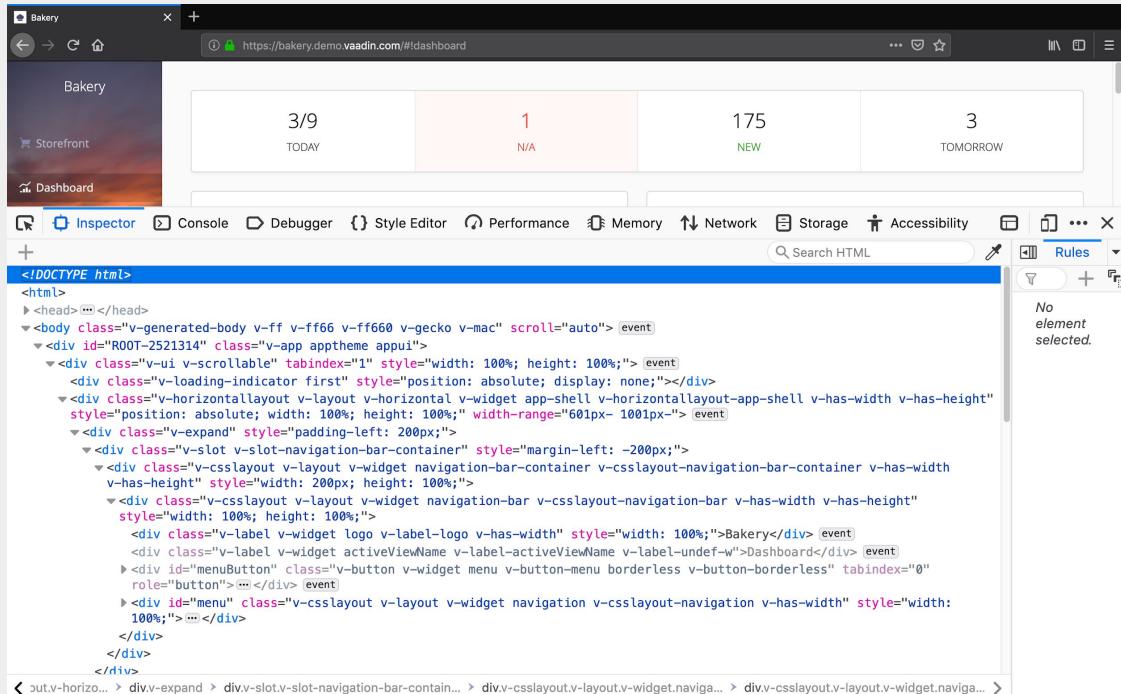
Allows to create new tags



Semantics

New tags for your browser

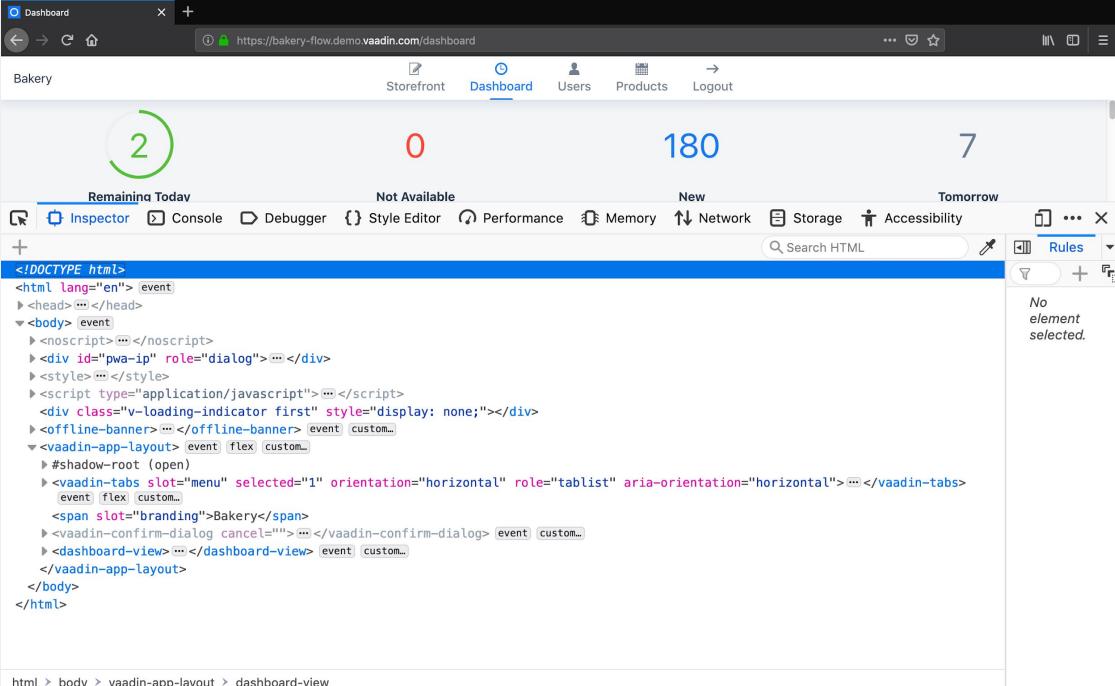
Component model for apps



Semantics

New tags for your browser

Component model for apps



The screenshot shows a browser window for 'Bakery' with the 'Dashboard' tab selected. The dashboard displays various metrics: 2 remaining today, 0 not available, 180 new, and 7 tomorrow. Below the dashboard, the browser's developer tools are open, specifically the 'Inspector' tab. The DOM tree is visible, showing the structure of the Vaadin application. The root node is <html>. The tree includes nodes like <head>, <body>, <div id="pwa-ip" role="dialog">, <script type="application/javascript">, <div class="v-loading-indicator first" style="display: none;">, <offline-banner>, <vaadin-app-layout>, <vaadin-tabs slot="menu" selected="1" orientation="horizontal" role="tablist" aria-orientation="horizontal">, Bakery, <vaadin-confirm-dialog cancel="!">, <dashboard-view>, and </body>. The bottom of the developer tools shows a breadcrumb navigation: html > body > vaadin-app-layout > dashboard-view. A status message in the bottom right corner of the developer tools panel says 'No element selected.'

True CSS Scoping

Shadow DOM

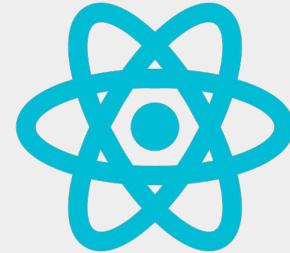
Current Solutions:

- Naming conventions: BEM, SMACSS
- Framework solutions: Angular, Vue
- CSS modules, CSS-in-JS

Reusability

Based on properties and DOM events

Framework agnostic



LitElement

Stability

Based on standards

Web standards live long



Use external components

With npm



Search custom elements

BUTTON

TOOLBAR

EMOJI

FORM

MEDIA

ROUTING

Browse elements

2037 Elements

**multiselect-combo-box**

A multi select combo box web component based on Polymer 2.x and the vaadin-combo-box

★ 11 2

**chemical-element-visualisation**

  A Web Component for visualizing an chemical element

★ 8 2

**actor-layout**

★ 1 0

**a-wc-router**

Routing Web Component

★ 4 0

**flip-card**

A Lit-Element component based on code-it notes by Dan Harding
https://www.instagram.com/same_dev_different_day/?hl=en

★ 0 0

**paper-pagination**

★ 0 0



Toggle



Toggle



Disabled



Invalid

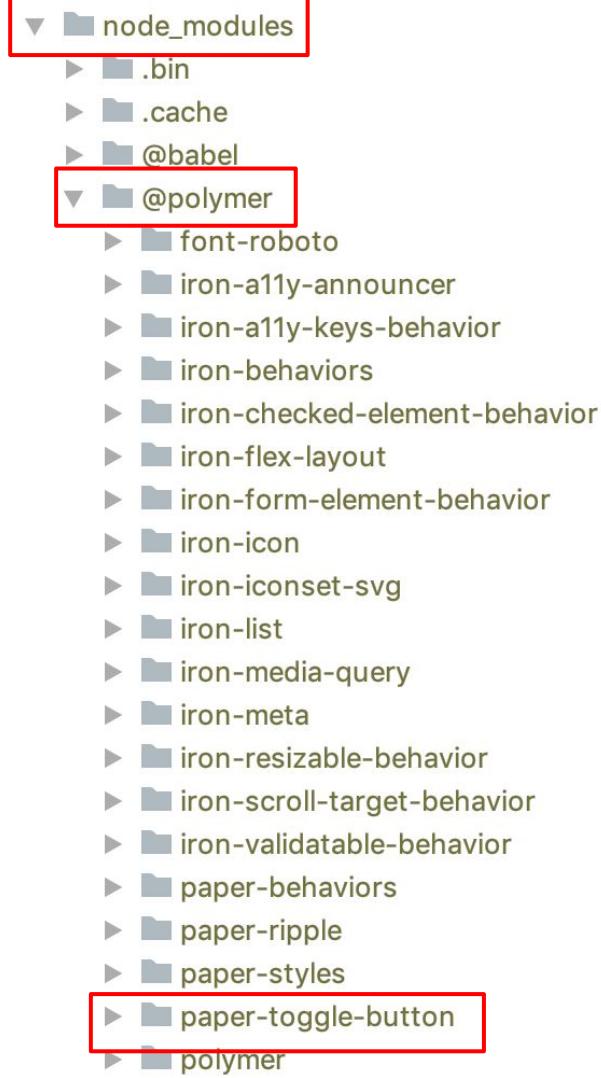
COPY

```
<paper-toggle-button>Toggle</paper-toggle-button>
<paper-toggle-button checked>Toggle</paper-toggle-button>
<paper-toggle-button disabled>Disabled</paper-toggle-button>
<paper-toggle-button invalid>Invalid</paper-toggle-button>
```

Install

Run npm install on the project's root directory

```
npm install --save @polymer/paper-toggle-button
```



Connect to Java

```
@Tag("paper-toggle-button")
@jsModule("@polymer/paper-toggle-button.js")
public class PaperToggleButton extends Component {
}
```

Use as if it were built-in

```
@Route
public class MainView extends VerticalLayout {

    public MainView() {
        add(new PaperToggleButton());
    }

}
```

Exercise

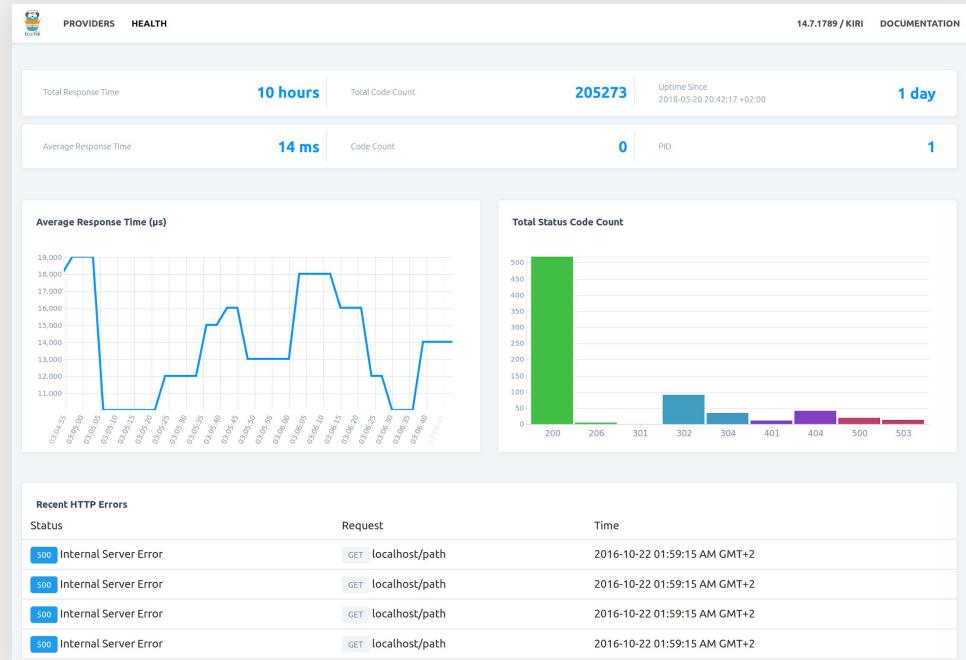
Tools

Tools

TestBench

Designer

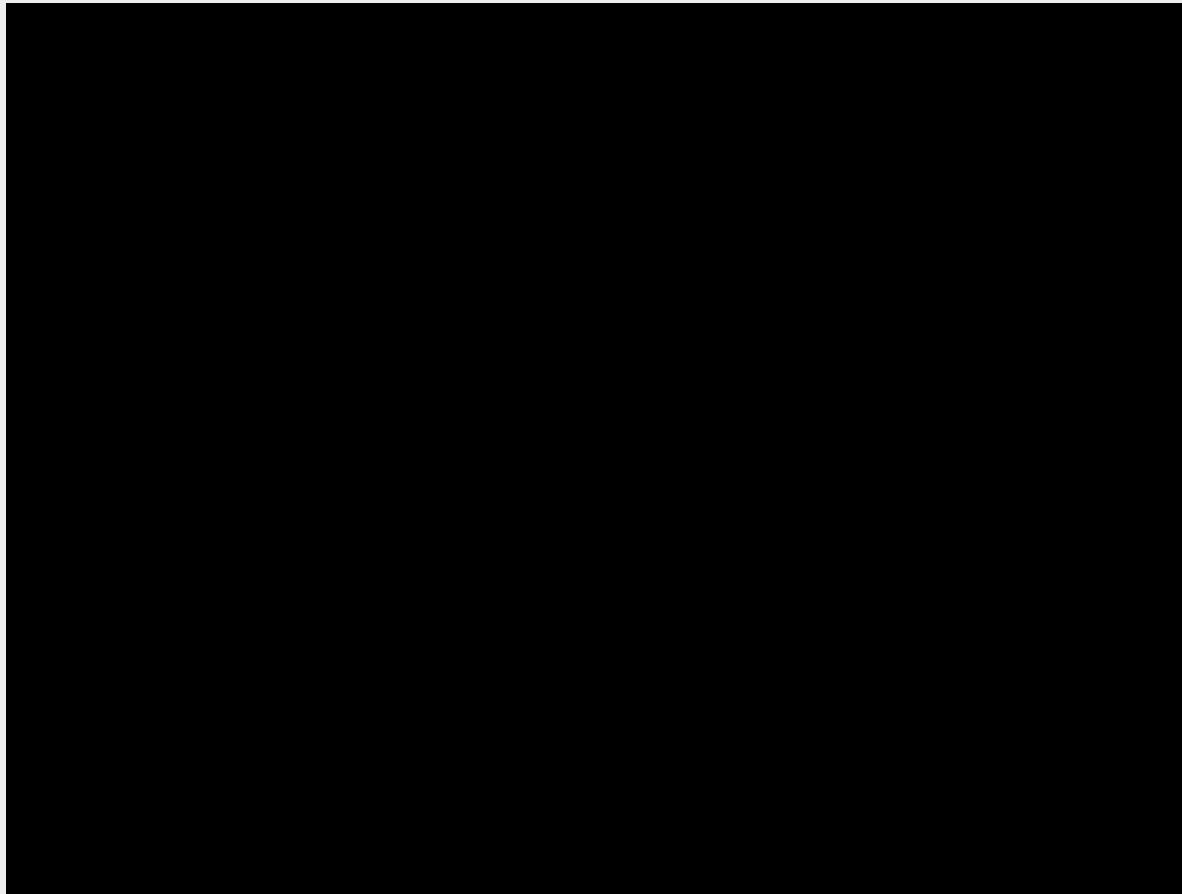
Multiplatform Runtime



What is TestBench?

Vaadin TestBench is a tool for creating and running browser-based integration tests for your Vaadin application

How does TestBench work?



Testbench vs Selenium

API for Vaadin Components

Testbench

```
//selects the row with index 5 in a grid  
grid.select(5);
```

Selenium

```
xpath=//div[@id='list']/div[3]/table/tbody/tr[2]/td
```

Server roundtrip

Testbench waits until the server-side is ready.

Selenium has to wait manually

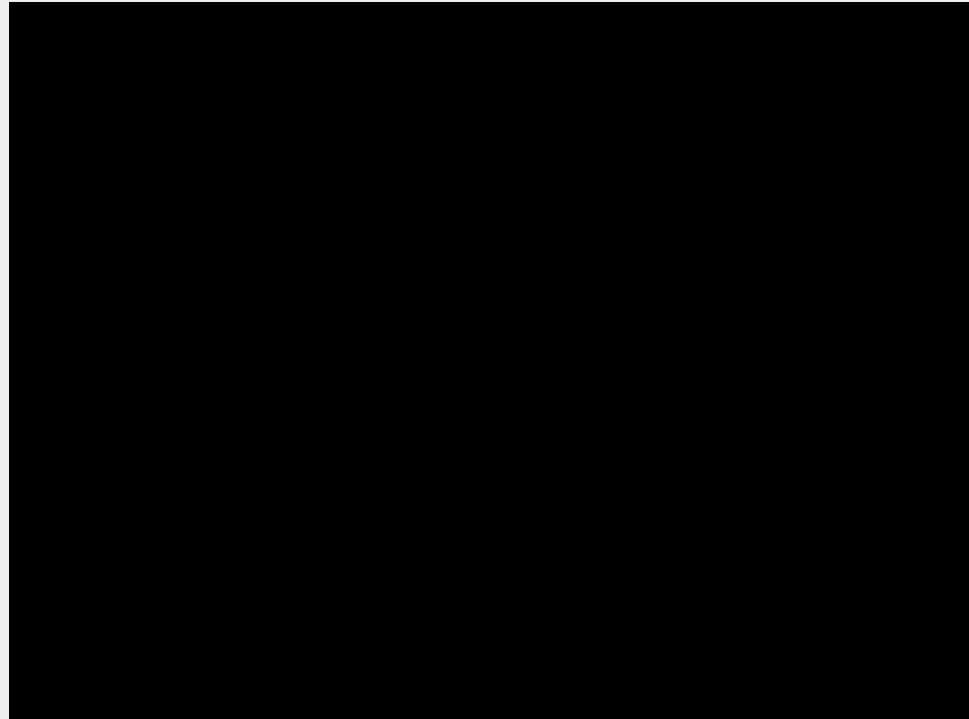
Shadow DOM

Vaadin TestBench has support for piercing Shadow DOM.

```
<!DOCTYPE html>
<html lang="en">
  >#shadow-root (open)
  ><head>...</head>
  ><body>
    ><identio-app page="identio-start-page">
      >#shadow-root (open)
        ><style scope="identio-app-0">...</style>
        ><iron-ajax auto url="https://identio-eb090.firebaseio.com/data.json" as="json" hidden></iron-ajax>
        ><app-location>...</app-location>
        ><app-route pattern="/:page"></app-route>
      ><app-drawer-layout fullbleed>
        >#shadow-root (open)
        ><app-drawer position="left" persistent opened style="touch-action: none;">
        ><app-header-layout has-scrolling-region id="appHeader">
          >#shadow-root (open)
          ><app-header condenses reveals effects="waterfall" style="border: 1px solid #000; border-radius: 10px; height: 40px; width: 100%; transition: border 0.3s ease; transform: translate3d(0px, 0px, 0px);">...</app-header>
          ><iron-pages role="main" attr-for-selected="name">
            >#shadow-root (open)
            ><identio-start-page name="identio-start-page" class="iron-page">
              >#shadow-root (open)
                ><style scope="identio-start-page">...</style>
```

Screen Comparison

When screen comparison fails, the differences are highlighted.



How does TestBench work?

Simulates a user of your application

performs the tasks specified using **Java code**

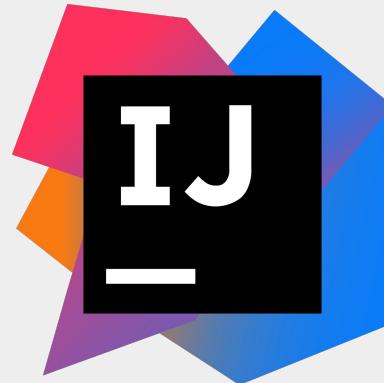
Verifies that the expected actions take place in the application.

The code

```
public class SimpleIT extends TestBenchTestCase {  
    ...  
  
    @Test  
    public void clickButton() {  
        // Find the first button (<vaadin-button>) on the page  
        ButtonElement button = $(ButtonElement.class).first();  
  
        // Click it  
        button.click();  
  
        Assert.assertTrue($(NotificationElement.class).exists());  
        Assert.assertEquals("Clicked", $(NotificationElement.class).first().getText());  
    }  
}
```

What is Vaadin Designer?

Designer is an IDE **plugin**



A **visual tool** for building Vaadin applications
with **Drag&Drop**

Viewport (571 x 759)

Search palette...

Patterns

- CRUD with Custom Col...
- CRUD with REST data
- Grid with REST data
- Form layout with Vaadin ele...
- Signup Form
- Tabs With Iron Pages

Project Components

- flow-component-renderer
- index
- reviews-list
- todo-view
- vaadin-grid-flow-selection-...

Components

POLYMER

- Dom Repeat
- Dom Repeat Elements

VAADIN BOARD

- Four Row Board

VAADIN BUTTON

- Icon Only Button
- Prefix Icon Button
- Primary Button
- Primary Success Button
- Primary Error Button
- Secondary Button
- Suffix Icon Button
- Tertiary Button

VAADIN CHARTS

- Area Chart
- Column Chart
- Line Chart
- Pie Chart

VAADIN CHECKBOX

- Checkbox with Label

Outline

Properties

Add Components

Drop to canvas

Drop elements here to start

- Form layout with Vaadin ele...
- Signup Form

Components

VAADIN TEXT FIELD

- Icon Text Field
- Prefixed Text Field
- Required Text Field
- RTL Text Field
- Suffix TextField
- Vaadin Text Area
- Vaadin Text Field

Parts

WEB COMPONENTS

- iron-autogrow-textarea
- paper-textarea
- vaadin-context-menu
- vaadin-context-menu-item
- vaadin-context-menu-overlay
- vaadin-grid-pro-edit-text-field
- vaadin-rich-text-editor
- vaadin-select-text-field
- vaadin-text-area
- vaadin-text-field
- vaadin-time-picker-text-field

HTML ELEMENTS

- input
- span

Aa Text

Add Components

Drop to the Outline view

The screenshot shows the Vaadin Design Studio interface. On the left is a large white canvas with the placeholder text "Drop elements here to start". To the right is a vertical component palette titled "Outline". The palette is divided into several sections:

- Patterns**:
 - Form layout with Vaadin ele...
 - Signup Form
- Components**:
 - VAADIN TEXT FIELD
 - Icon Text Field
 - Prefixed Text Field
 - Required Text Field
 - RTL Text Field
 - Suffix TextField
 - Vaadin Text Area
 - Vaadin Text Field
- Parts**:
 - WEB COMPONENTS
 - iron-autogrow-textarea
 - paper-textarea
 - vaadin-context-menu
 - vaadin-context-menu-item
 - vaadin-context-menu-overlay
 - vaadin-grid-pro-edit-text-field
 - vaadin-rich-text-editor
 - vaadin-select-text-field
 - vaadin-text-area
 - vaadin-text-field
 - vaadin-time-picker-text-field
- HTML ELEMENTS**:
 - input
 - span

A mouse cursor is hovering over the "text" component in the "HTML ELEMENTS" section. The "Properties" panel is visible on the far right.

Patterns

Quick-start solutions to certain design tasks

Patterns

- CRUD with Custom Columns
- CRUD with REST data
- Grid with REST data
- Form layout with Vaadin ele...
- Signup Form
- Tabs With Iron Pages



Patterns

Grid with REST data

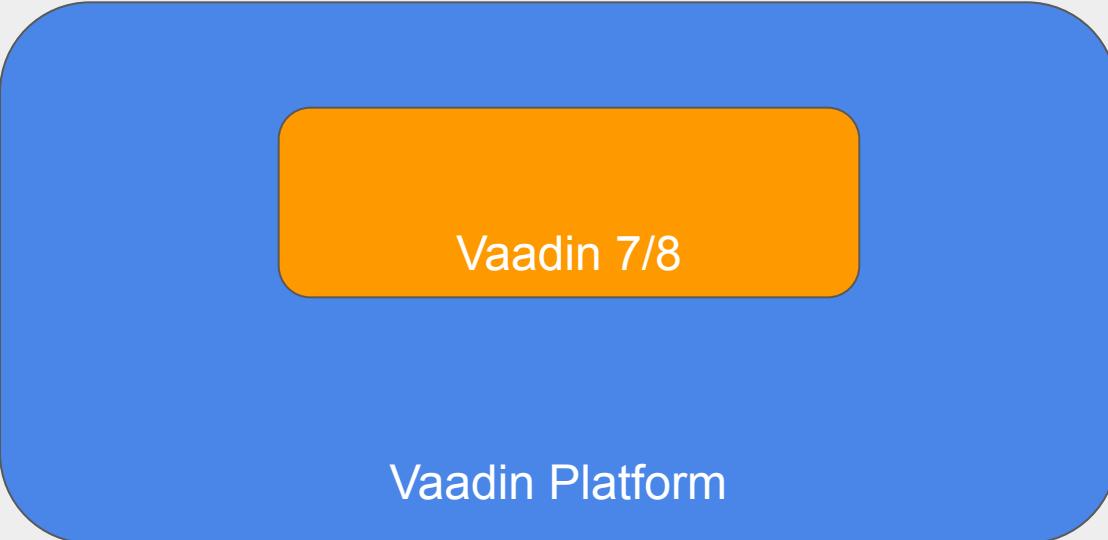
#	First Name	Last Name	Address
0	Luis	Morales	7009 Sunny Elk By-pass, Herculanum
1	Katherine	Sanchez	863 Silent Horse Street, Novice
2	Jasmine	Perry	7664 Cotton Leaf Pathway, Skamokawa
3	Hailey	Hernandez	1392 Silver Lagoon Pointe, Cheektowasa
4	Layla	Smith	9874 Quiet Isle, Paydown
5	Hadley	Myers	8157 Silver Heath, Steam Corners

Patterns

CRUD with custom columns

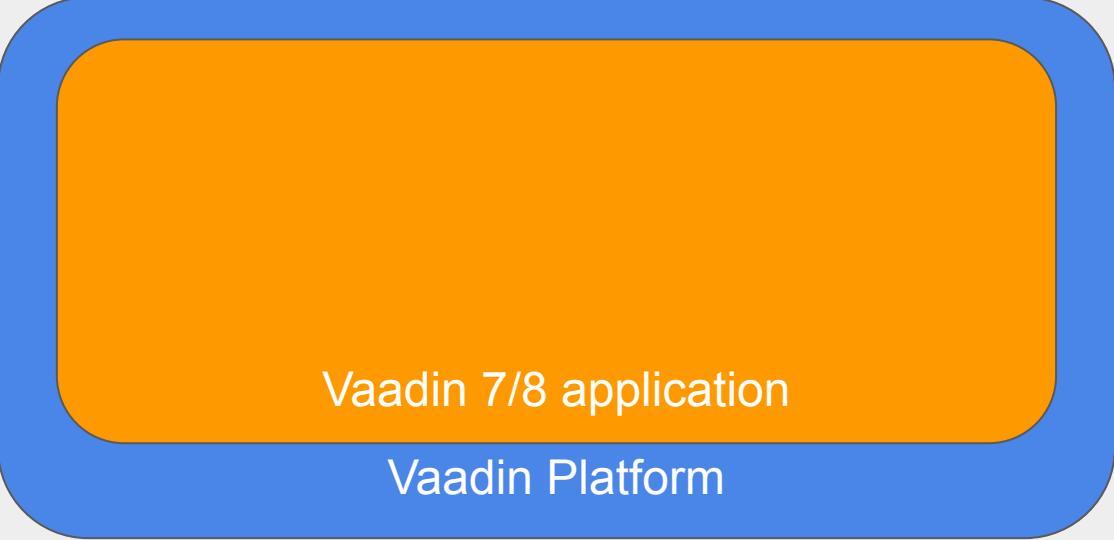
User	Email
 	amelia jones amelia.jones@example.com
 	mercedes gil mercedes.gil@example.com
 	xavier blanco xavier.blanco@example.com
 	zoe holland zoe.holland@example.com
 	sofia bouchard sofia.bouchard@example.com
	
	<a data-bbox="1728 764 1864 791" href="#">New item

Multiplatform Runtime



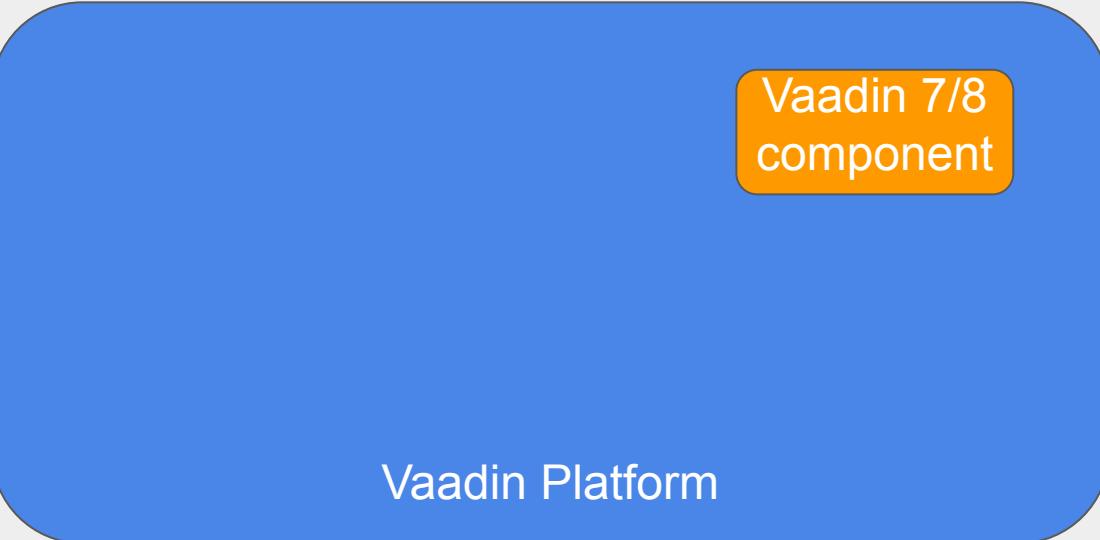
Vaadin 7/8

Vaadin Platform



Vaadin 7/8 application

Vaadin Platform



Vaadin 7/8
component

Vaadin Platform

How does it work?

Cannot use legacy components directly in the new application

```
Button legacyButton = new Button("Legacy button");
VerticalLayout flowLayout = new VerticalLayout();
flowLayout.add(legacyButton);
```

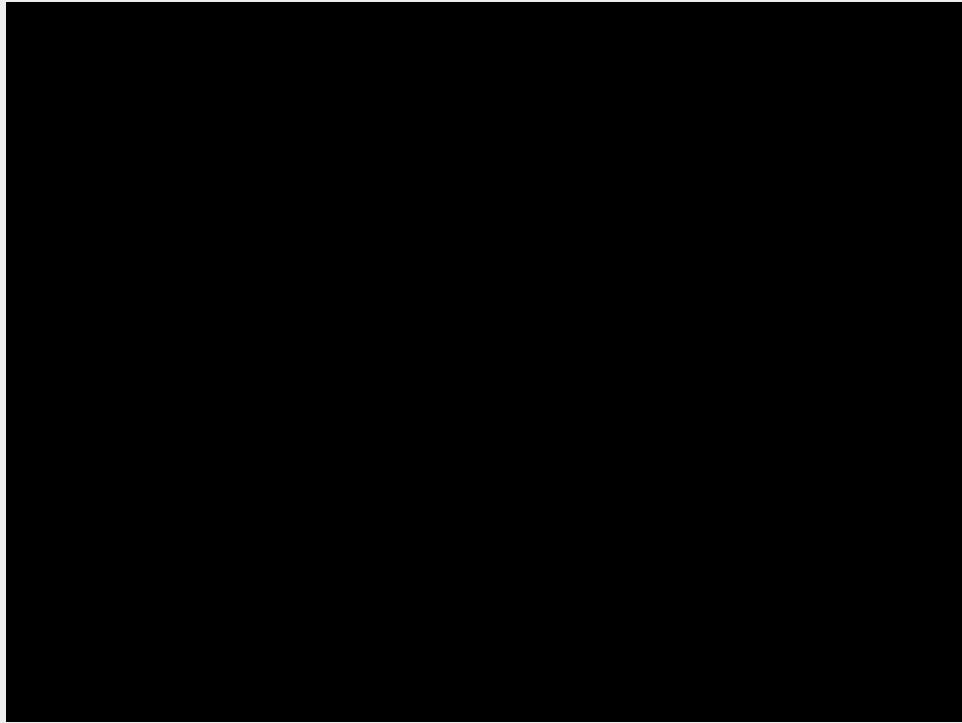
How does it work?

Wrap legacy component with a legacy wrapper

```
Button legacyButton = new Button("Legacy button");
VerticalLayout flowLayout = new VerticalLayout();
flowLayout.add(new LegacyWrapper(legacyButton));
```

Progressive Web Apps

Web apps with native-like experiences



PWA How

Web App Manifest

Service Worker

Icons

Offline support

Header information

Installation prompt

Create PWA with Vaadin

Just by adding @PWA, a Vaadin application becomes a PWA

```
@Route
@PWA(name = "My Progressive Web Application", shortName = "MyPWA")
public class MainView extends VerticalLayout {
    public MainView() {
        Button button = new Button("Click me",
            e -> Notification.show("Hello Worlld!"));
        add(button);
    }
}
```

Summary

- What is Vaadin
- Java
- Components
- Tools
- Progressive Web Application

Feedback

bit.ly/vaadin-training