



Devoxx Antwerpen

Marcel Schutte - Groningen - 3 november 2023

Wie ben ik?

- **Functie** - Lead Solution Architect
- **Opdracht** - KOOP in Den Haag
- **Intern** - Lid van Java Architecture Board,
Java Education Team en Technology Board
NoordOost

Persoonlijk

- **Leeftijd** 41 jaar
- **Burgerlijke staat** Getrouwd, 2 kinderen
- **Hobby's** Hardlopen, geocachen, tuinieren





DEVCON

"Where AI and mankind shall merge as One" #TomorrowIsNow



Delen



Bekijken op YouTube

<https://www.youtube.com/watch?v=njs7bq0oftg>

Wat is Devoxx?

- Grootste Java conferentie van Europa(de wereld?)
- Jaarlijks evenement over Java en veel aanverwante technieken/methodieken
- Honderden sessies en sprekers

Daarnaast:

- Bijpraten met oudcollega's
- Netwerken in de verschillende bedrijfsstands
- Huidige collega's beter leren kennen
- Heel veel energie om nieuwe dingen uit te proberen.



Agenda

1. Java 21
2. Ondersteuning voor VS Code
3. Sustainability
4. AI assistants
5. Diverse interessante sessies



Java 21

1. Toegankelijkheid ➡
2. Virtual Threads
3. Verschillende extra features

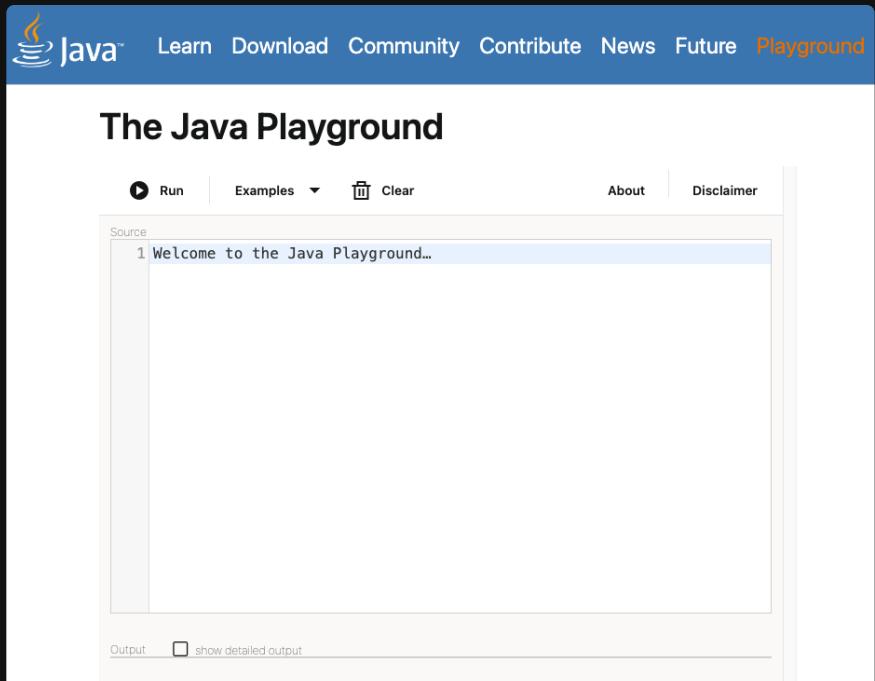
Unnamed classes & Instance main methods

```
public class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello, World!");  
    }  
  
    void main() {  
        System.out.println("Hello, World!");  
    }  
}
```

Java 21

1. Toegankelijkheid ➡
2. Virtual Threads
3. Verschillende extra features

Java Playground at Dev.java



<https://dev.java/playground>

Java 21

1. Toegankelijkheid
2. Virtual Threads👉
3. Verschillende extra features

Project Loom

Virtual threads

- Lighter threads
- min. size -> 200-300 bytes
- schaalt naar 1M+ threads op doorsnee hardware

Virtual Threads are real Threads

- implementeren `java.lang.Thread` en ondersteunen `ThreadLocal`
- clean stacktraces
- sequential step debugging
- threaded code just works
- Thread without the bagage

Java 21

1. Toegankelijkheid
2. Virtual Threads👉
3. Verschillende extra features

Virtual threads (JEP 444)

```
try (var executor =  
      Executors.newVirtualThreadPerTaskExecutor()) {  
    IntStream.range(0, 10_000).forEach(i -> {  
        executor.submit(() -> {  
            Thread.sleep(Duration.ofSeconds(1));  
            return i;  
        });  
    });  
} // executor.close() is called implicitly, and waits
```

Java 21

1. Toegankelijkheid
2. Virtual Threads
3. Verschillende extra features👉

String templates

```
// Embedded expressions can be strings
String firstName = "Bill";
String lastName  = "Duck";
String fullName  = STR."\{firstName} \{lastName}";
| "Bill Duck"
String sortName  = STR."\{lastName}, \{firstName}";
| "Duck, Bill"

// Embedded expressions can perform arithmetic
int x = 10, y = 20;
String s = STR."\{x} + \{y} = \{x + y}";
| "10 + 20 = 30"

// Embedded expressions can invoke methods/access fields
String s =
    STR."You have a \{getOfferType()} waiting for you!";
| "You have a gift waiting for you!"
```

Java 21

1. Toegankelijkheid
2. Virtual Threads
3. Verschillende extra features👉

Pattern matching for switch

```
// As of Java 21
static String formatterPatternSwitch(Object obj) {
    return switch (obj) {
        case Integer i → String.format("int %d", i);
        case Long l    → String.format("long %d", l);
        case Double d  → String.format("double %f", d);
        case String s   → String.format("String %s", s);
        default         → obj.toString();
    };
}
```

Java 21

1. Toegankelijkheid
2. Virtual Threads
3. Verschillende extra features👉

Switches and null

```
// Prior to Java 21
static void testFooBarOld(String s) {
    if (s == null) {
        System.out.println("Oops!");
        return;
    }
    switch (s) {
        case "Foo", "Bar" → System.out.println("Great");
        default           → System.out.println("Ok");
    }
}

// As of Java 21
static void testFooBarNew(String s) {
    switch (s) {
        case null         → System.out.println("Oops");
        case "Foo", "Bar" → System.out.println("Great");
        default          → System.out.println("Ok");
    }
}
```

Java 21

1. Toegankelijkheid
2. Virtual Threads
3. Verschillende extra features👉

Case refinement

```
// Prior to Java 21
static void testStringOld(String response) {
    switch (response) {
        case String s → {
            if (s.equalsIgnoreCase("YES"))
                System.out.println("You got it");
            else
                System.out.println("Sorry?");
        }
    }
}

// As of Java 21
static void testStringNew(String response) {
    switch (response) {
        case String s
        when s.equalsIgnoreCase("YES") → {
            System.out.println("You got it");
        }
        case String s → {
            System.out.println("Sorry?");
        }
    }
}
```

Visual Studio Code

Oracle Java Platform Extension for Visual Studio Code

Features

- project view
- auto-complete
- error highlighting
- jump to definition
- some forms of automated refactoring
- JavaDoc-on-hover
- debugging support
- unit-testing support for JUnit
- support for Gradle and Maven projects.



Backspace

Home

4



Sustainability certificering

THE LINUX FOUNDATION



Catalog ▾ | Resources ▾ | Corporate Solutions | Explore ▾

[Training](#) > [Open Source Best Practice](#) > Green Software for Practitioners (LFC131)

TRAINING COURSE

Green Software for Practitioners (LFC131)

Learn the basic concepts a software practitioner needs to know to build, maintain and run greener applications.
[Course Rating](#)

4.5/5 Stars

<https://training.linuxfoundation.org/training/green-software-for-practitioners-lfc131/>

AI Coding assistants

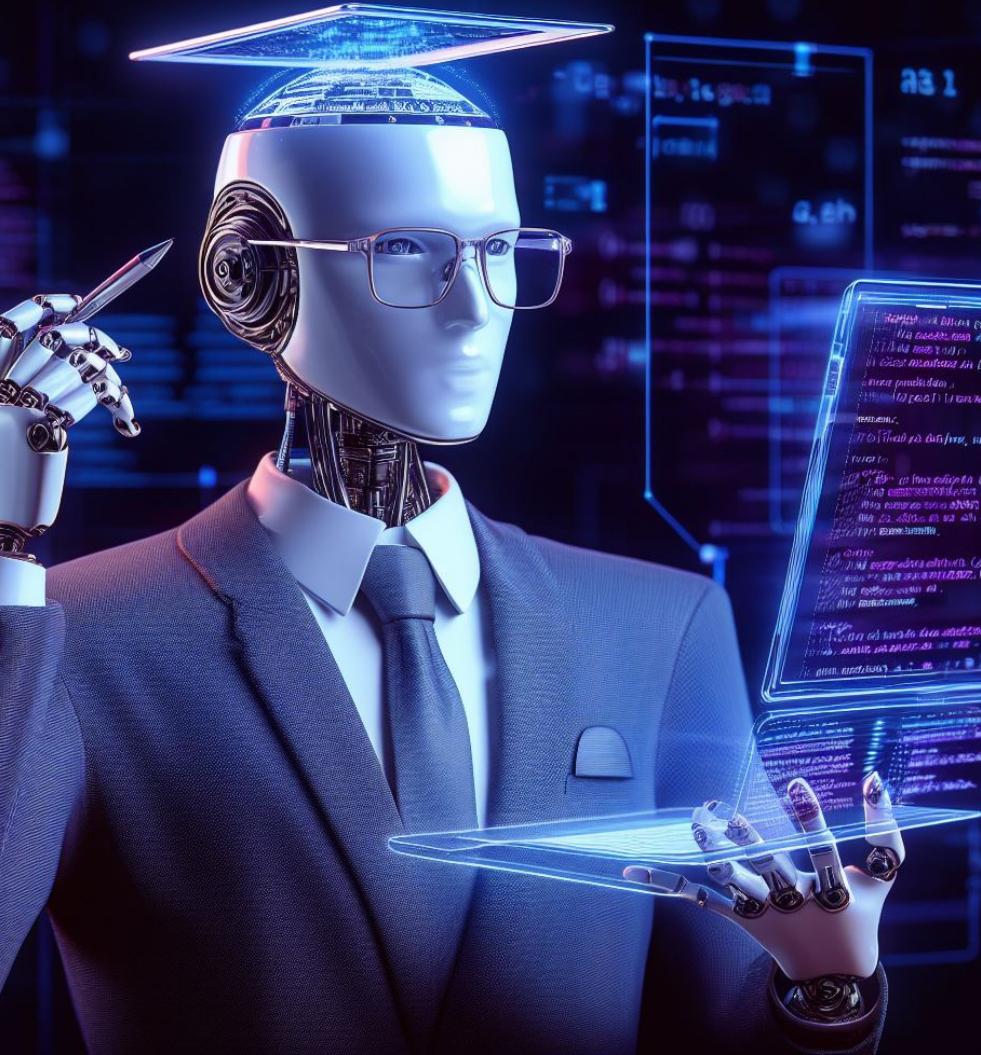
Programmeren zal nooit meer hetzelfde zijn

Nu al beschikbaar

- Amazon CodeWhisperer (Gratis)
- Github Copilot
- Tabnine

Binnenkort beschikbaar

- Github Copilot Chat
- Tabnine Chat
- JetBrains AI assistant





JetBrains AI Assistant

Technical preview with limited access

Functionaliteit

- AI Chat
- documentatie/commentaar genereren
- commit messages genereren
- code uitleggen
- refactoren
- bugfixes voorstellen
- testen schrijven

Imposter Syndrome

Wie?

ledereen!

Is dat een probleem?

Nee!

Embracing Imposter Syndrome

By Dom Hodgson

<https://devoxx.be/talk/?id=70101>



C4 models as Code

Voordelen:

- meerdere diagrammen uit 1 model
- alternatieve visualisaties
- versiebeheer

C4 models as code

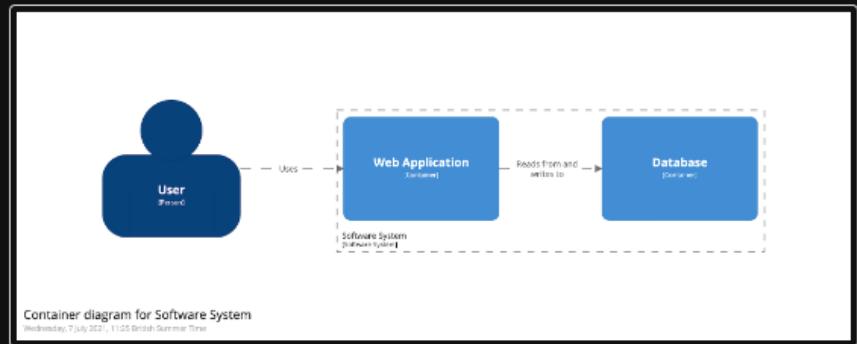
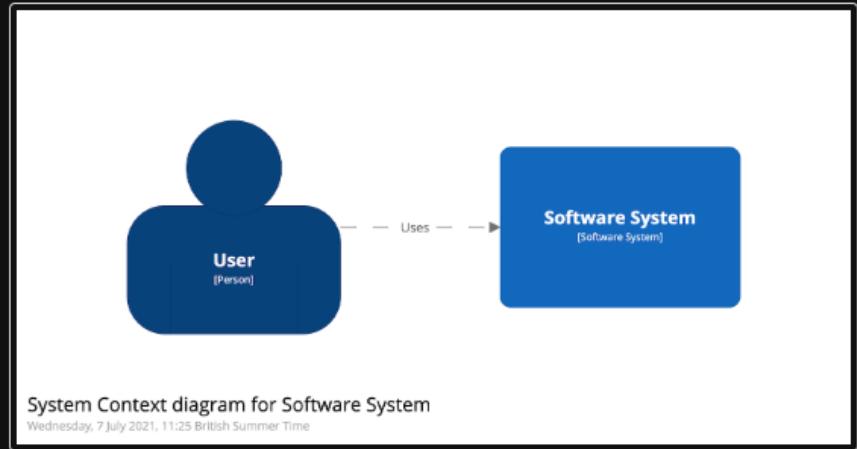
By Simon Brown

<https://devoxx.be/talk/?id=28357>

```
workspace {
    model {
        user = person "User"
        softwareSystem = softwareSystem "SW System" {
            webapp = container "Web Application" {
                user → this "Uses"
            }
            container "Database" {
                webapp → this "Reads from and writes to"
            }
        }
        views {
            systemContext softwareSystem {
                include *
                autolayout lr
            }
            container softwareSystem {
                include *
                autolayout lr
            }
        }
    }
}
```

C4 models as Code

```
workspace {  
    model {  
        user = person "User"  
        softwareSystem = softwareSystem "SW System" {  
            webapp = container "Web Application" {  
                user → this "Uses"  
            }  
            container "Database" {  
                webapp → this "Reads from and writes to"  
            }  
        }  
        views {  
            systemContext softwareSystem {  
                include *  
                autolayout lr  
            }  
            container softwareSystem {  
                include *  
                autolayout lr  
            }  
        }  
    }  
}
```



Building a Bullsh*t Language

Annotationscript

Wat?

- gebaseerd op java annotaties
- turing compleet

Waarom?

- Waarom niet?
- Omdat het kan!

```
@Zero("begin")
@Zero(list={
    @One("define"),
    @One("twice"),
    @One(list={
        @Two("lambda"),
        @Two(list={@Three("x")}),
        @Two(list={@Three("*"), @Three("x"), @Three("2")})}))})
@Zero(list={
    @One("+"),
    @One(list=[@Two("twice"), @Two("x")]),
    @One("1")})
class Example {}
```

Building a Bullsh*t Language

By Jan Ouwens

<https://devoxx.be/talk/?id=31405>

Testing microservices

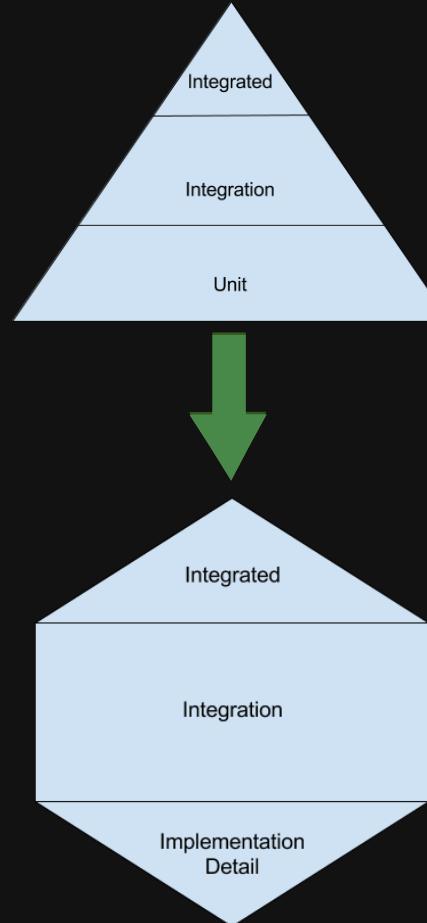
Integration testing challenges

- Test Coupling
- Flaky tests
- Slow tests
- Cognitive load
- Investment -> CI, libs, training

Testing Microservices - Join the Revolution

By Victor Rentea

<https://devoxx.be/talk/?id=30503>



Tests need architecting

- Naming conventions
- Object Mother

Your tests also need some architecting
By Jonas Geiregat

<https://devoxx.be/talk/?id=2914>

```
Address billingAddress = new Address.Builder()  
    .streetAddress("123 Main St")  
    .city("Anytown")  
    .state("CA")  
    .zipCode("12345")  
    .country(Country.US)  
    .build();  
  
Address shippingAddress = new Address.Builder()  
    .streetAddress("456 Oak Ave")  
    .city("Othertown")  
    .state("CA")  
    .zipCode("67890")  
    .country(Country.US)  
    .build();  
  
List<InvoiceItem> items = new ArrayList<>();  
items.add(new InvoiceItem("Product A", Amount.EUR(100.00),  
    items.add(new InvoiceItem("Product B", Amount.EUR(100.00),  
Customer customer = new Customer.Builder()  
    .name("John Doe")  
    .email("john.doe@example.com")  
    .phoneNumber("555-123-4567")  
    .build();  
  
Invoice invoice = new Invoice(  
    new InvoiceNumber("001"),  
    customer,  
    billingAddress,  
    shippingAddress,  
    LocalDate.now(),
```

Tests need architecting

```
Address billingAddress = new Address.Builder()
    .streetAddress("123 Main St")
    .city("Anytown")
    .state("CA")
    .zipCode("12345")
    .country(Country.US)
    .build();

Address shippingAddress = new Address.Builder()
    .streetAddress("456 Oak Ave")
    .city("Othertown")
    .state("CA")
    .zipCode("67890")
    .country(Country.US)
    .build();

List<InvoiceItem> items = new ArrayList<>();
items.add(new InvoiceItem("Product A", Amount.EUR(100.00), Tax.vatPercentage(21)));
items.add(new InvoiceItem("Product B", Amount.EUR(100.00), Tax.vatPercentage(21)));
Customer customer = new Customer.Builder()
    .name("John Doe")
    .email("john.doe@example.com")
    .phoneNumber("555-123-4567")
    .build();

Invoice invoice = new Invoice(
    new InvoiceNumber("001"),
    customer,
```

Tests need architecting

```
public class InvoiceMother {  
  
    private InvoiceMother() { }  
  
    public static Builder invoice() {  
        return new Builder();  
    }  
  
    public static class Builder {  
  
        InvoiceNumber invoiceNumber = new InvoiceNumber("001");  
        Customer customer = CustomerMother.customer().build();  
        Address billingAddress = AddressMother.address().build();  
        Address shippingAddress = AddressMother.address().build();  
        LocalDate creationDate = LocalDate.now();  
        List<InvoiceItem> items = List.of(InvoiceItemMother.item().build());  
  
        public Builder withItems(List<InvoiceItem> items) {  
            this.items = items;  
            return this;  
        }  
  
        // other setters are left out for brevity  
  
        public Invoice build() {
```

Tests need architecting

```
Invoice invoice = InvoiceMother.invoice()
    .withInvoiceItems(
        new InvoiceItem("Product A", Amount.EUR(100.00), Tax.vatPercentage(21)),
        new InvoiceItem("Product B", Amount.EUR(100.00), Tax.vatPercentage(21)))
    .build();

Amount amount = invoice.getVatAmount();

assertThat(amount).isEqualTo(Amount.EUR(42))
```



SoCode 2023

Jaarlijkse programmeerwedstrijd

- Start op 1 december, 6.00 uur
- 2 programmeerpuzzels per dag

Extra's

- Discordserver voor discussie
- Ranglijst specifiek voor deelnemers van Sogeti en CapGemini
- Mooie prijzen!

Kickoff: maandag 27 november om 16:00

Aanmelden:

<https://forms.microsoft.com/e/u8zxgvT0fW>





Bedankt voor jullie aandacht!

 [linkedin.com/in/marceljschutte](https://www.linkedin.com/in/marceljschutte)

 @PragArchitect

 marcel.schutte@sogeti.com

 +31 652 659 277



<https://github.com/marceljschutte/talks/tree/main/java-friday-pizza-devoxx>

