Götterdämmerung

Valhala, Panama, Loom und Co.

Neuerungen bei Java 18, 19 etc.

Michael Wiedeking, MATHEMA GmbH

Was Sie erwartet

- Java 18
- Java 19
- Dies und das

- Zu wenig von allem

Was Sie nicht erwartet

- 11:10 - 11:55 Uhr
  Javas neue Gesprächskultur: ganz wie in Panama
  Bernd Müller

- 12:15 - 13:00 Uhr
  Pattern Matching in Java
  Falk Sippach

- 14:30 - 15:15 Uhr
  Virtual Threads
  Jörg Hettel

- 15:35 - 16:20 Uhr
  Jewel Java
  Merlin Bögershausen

Java 18

- ~~400: UTF-8 by Default~~
- 408: Simple Web Server
- 413: Code Snippets in Java API Documentation
- ~~416: Reimplement Core Reflection with Method Handles~~
- 417: Vector API (Third Incubator)
- ~~418: Internet-Address Resolution SPI~~
- 419: Foreign Function & Memory API (Second Incubator)
- 420: Pattern Matching for switch (Second Preview)
- 421: Deprecate Finalization for Removal

Java 18

- 408: Simple Web Server
- 413: Code Snippets in Java API Documentation
- 417: Vector API (Third Incubator)
- 419: Foreign Function & Memory API (Second Incubator)
- 420: Pattern Matching for switch (Second Preview)
- 421: Deprecate Finalization for Removal

Java 18

- **408: Simple Web Server**
- 413: Code Snippets in Java API Documentation
- 417: Vector API (Third Incubator)
- 419: Foreign Function & Memory API (Second Incubator)
- 420: Pattern Matching for switch (Second Preview)
- 421: Deprecate Finalization for Removal

```
$ jwebserver
```

```
$ jwebserver

Binding to loopback by default. For all interfaces use "-b
0.0.0.0" or "-b ::".

Serving /cwd and subdirectories on 127.0.0.1 port 8000

URL: http://127.0.0.1:8000/
```

```
$ jwebserver -h
Options:
       -h or -? or --help
              Prints the help message and exits.
       -b addr or --bind-address addr
              Specifies the address to bind to.  Default: 127.0.0.1 or ::1 (loopback).  For
              all interfaces use -b 0.0.0.0 or -b ::.
       -d dir or --directory dir
              Specifies the directory to serve.  Default: current directory.
       -o level or --output level
              Specifies the output format.  none | info | verbose.  Default: info.
       -p port or --port port
              Specifies the port to listen on.  Default: 8000.
       -version or --version
              Prints the version information and exits.
       To stop the server, press Ctrl + C.
$
```

Java 18

- 408: Simple Web Server
- **413: Code Snippets in Java API Documentation**
- 417: Vector API (Third Incubator)
- 419: Foreign Function & Memory API (Second Incubator)
- 420: Pattern Matching for switch (Second Preview)
- 421: Deprecate Finalization for Removal

```
/**
 * A simple program.
 * {@snippet :
 * class HelloWorld {
 *    public static void main(String... args) {
 *       System.out.println("Hello World!"); // @highlight substring="println"
 *     }
 * }
 * }
 */
```

```
A simple program.

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

```
/**
 * {@snippet :
 *   public static void main(String... args) {
 *       for (var arg : args) {                  // @highlight region regex = "\barg\b"
 *           if (!arg.isBlank()) {
 *               System.out.println(arg);
 *           }
 *       }                                       // @end
 *   }
 *   }
 */
```

```
public static void main(String... args) {
    for (var arg : args) {
        if (!arg.isBlank()) {
            System.out.println(arg);
        }
    }
}
```

```
{@snippet file=external-snippets.properties region=house2 }
```

Datei: `./snippet-files/external-snippets.properties`

```
...
# @start region=house2
house.number=42
# @highlight substring="Main St." :
house.street=Main St.
house.town=AnyTown, USA
# @end region=house2
...
```

```
house.number=42

house.street=Main St.

house.town=AnyTown, USA
```

Java 18

- 408: Simple Web Server
- 413: Code Snippets in Java API Documentation
- **417: Vector API (Third Incubator)**
- 419: Foreign Function & Memory API (Second Incubator)
- 420: Pattern Matching for switch (Second Preview)
- 421: Deprecate Finalization for Removal

```
void scalarComputation(float[] a, float[] b, float[] c) {
   for (int i = 0; i < a.length; i++) {
        c[i] = (a[i] * a[i] + b[i] * b[i]) * -1.0f;
   }
}
```

```
void scalarComputation(float[] a, float[] b, float[] c) {
   for (int i = 0; i < a.length; i++) {
       c[i] = (a[i] * a[i] + b[i] * b[i]) * -1.0f;
   }
}
```

Registerbreite (Bit)

	64	128	256	512
byte	8	16	32	64
short	4	8	16	32
int	2	4	8	16
long	1	2	4	8
float	2	4	8	16
double	1	2	4	8

Anzahl der Elemente des gegebenen Typs

```
static final VectorSpecies<Float> SPECIES = FloatVector.SPECIES_PREFERRED;

void vectorComputation(float[] a, float[] b, float[] c) {
    int i = 0;
    int upperBound = SPECIES.loopBound(a.length);
    for (; i < upperBound; i += SPECIES.length()) {
        var va = FloatVector.fromArray(SPECIES, a, i);
        var vb = FloatVector.fromArray(SPECIES, b, i);
        var vc = va.mul(va).add(vb.mul(vb)).neg();
        vc.intoArray(c, i);
    }
}
```

```
for (; i < a.length; i++) {
    c[i] = (a[i] * a[i] + b[i] * b[i]) * -1.0f;
}
```

- VectorOperators.Associative
- VectorOperators.Binary
- VectorOperators.Comparison
- VectorOperators.Conversion<E,F>
- VectorOperators.Ternary
- VectorOperators.Test
- VectorOperators.Unary

- ASHR
- ASIN
- ATAN
- ATAN2
- B2D
- B2F
- B2I
- B2L
- B2S
- BITWISE_BLEND
- CBRT
- COS
- COSH
- D2B
- D2F
- D2I
- D2L
- D2S
- DIV
- EQ
- EXP
- EXPM1
- F2B
- F2D
- F2I
- F2L
- F2S
- FIRST_NONZERO
- FMA
- GE
- GT
- HYPOT
- I2B
- I2D
- I2F
- I2L
- I2S
- IS_DEFAULT
- IS_FINITE
- IS_INFINITE
- IS_NAN
- IS_NEGATIVE
- L2B
- L2D
- L2F
- L2I
- L2S
- LE
- LOG
- LOG10
- LOG1P
- LSHL
- LSHR
- LT
- MAX
- MIN
- MUL
- NE
- NEG
- NOT
- OR
- POW
- REINTERPRET_D2L
- REINTERPRET_F2I
- REINTERPRET_I2F
- REINTERPRET_L2D
- ROL
- ROR
- S2B
- S2D
- S2F
- S2I
- S2L
- SIN
- SINH
- SQRT
- SUB
- TAN
- TANH
- UNSIGNED_GE
- UNSIGNED_GT
- UNSIGNED_LE
- UNSIGNED_LT
- XOR
- ZERO_EXTEND_B2I
- ZERO_EXTEND_B2L
- ZERO_EXTEND_B2S
- ZERO_EXTEND_I2L
- ZERO_EXTEND_S2I
- ZERO_EXTEND_S2L
- ZOMO

Java 18

- 408: Simple Web Server
- 413: Code Snippets in Java API Documentation
- 417: Vector API (Third Incubator)
- **419: Foreign Function & Memory API (Second Incubator)**
- 420: Pattern Matching for switch (Second Preview)
- 421: Deprecate Finalization for Removal

- Die Anzahl der zu dereferenzierenden Bytes
- Die Einschränkungen bei der Adresse, über die die Dereferenzierung erfolgt
- Die Byte-Ordnung des betroffenen Bereichs
- Der Java-Typ, der bei der Dereferenzierung geliefert werden soll (int vs. float)

```
MemorySegment segment = MemorySegment.allocateNative(
    100,
    MemorySession.openImplicit()
);

for (int i = 0; i < 25; i++) {

    segment.setAtIndex(
        ValueLayout.JAVA_INT,
        i,      // Index.
        i       // Zu schreibender Wert.
    );

}
```

```
struct Point {
   int x;
   int y;
} pts[10];

MemorySegment segment = MemorySegment.allocateNative(
   2 * 4 * 10,
   MemorySession.openImplicit()
);

for (int i = 0; i < 10; i++) {
   segment.setAtIndex(ValueLayout.JAVA_INT, i, vx(i));              // x
   segment.setAtIndex(ValueLayout.JAVA_INT, (i * 2) + 1, vy(i));    // y
}
```

```
SequenceLayout ptsLayout = MemoryLayout.sequenceLayout(
    10,
    MemoryLayout.structLayout(
        ValueLayout.JAVA_INT.withName("x"),
        ValueLayout.JAVA_INT.withName("y")
    )
);
```

```
VarHandle xHandle;                      // (MemorySegment, long) –> int
xHandle = ptsLayout.varHandle(
    PathElement.sequenceElement(),
    PathElement.groupElement("x")
);

VarHandle yHandle;                      // (MemorySegment, long) –> int
yHandle = ptsLayout.varHandle(
    PathElement.sequenceElement(),
    PathElement.groupElement("y")
);
```

```
MemorySegment segment = MemorySegment.allocateNative(
    ptsLayout,
    MemorySession.openImplicit()
);

for (int i = 0; i < ptsLayout.elementCount().getAsLong(); i++) {
    xHandle.set(segment, (long) i, vx(i));
    yHandle.set(segment, (long) i, vy(i));
}
```

Java 18

- 408: Simple Web Server
- 413: Code Snippets in Java API Documentation
- 417: Vector API (Third Incubator)
- 419: Foreign Function & Memory API (Second Incubator)
- **420: Pattern Matching for switch (Second Preview)**
- 421: Deprecate Finalization for Removal

```
static String formatter(Object o) {
    String formatted = "unknown";
    if (o instanceof Integer i) {
        formatted = String.format("int %d", i);
    } else if (o instanceof Long l) {
        formatted = String.format("long %d", l);
    } else if (o instanceof Double d) {
        formatted = String.format("double %f", d);
    } else if (o instanceof String s) {
        formatted = String.format("String %s", s);
    }
    return formatted;
}
```

```
static String formatter(Object o) {
    return switch (o) {
        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 ->
            "unknown";
    };
}
```

```
static void testTriangle(Shape s) {
    switch (s) {
    case null:
        break;
    case Triangle t:
        if (t.calculateArea() > 100) {
            System.out.println("Large triangle");
            break;
        }
    default:
        System.out.println("A shape, possibly a small triangle");
    }
}
```

```
static void testTriangle(Shape s) {
    switch (s) {
    case Triangle t when t.calculateArea() > 100 ->
        System.out.println("Large triangle");
    default ->
        System.out.println("A shape, possibly a small triangle");
    }
}
```

```
static void testTriangle(Shape s) {

    switch (s) {
    case Triangle t when t.calculateArea() > 100 ->
        System.out.println("Large triangle");
    case Triangle t ->
        System.out.println("Small triangle");
    default ->
        System.out.println("Non-triangle");
    }

}
```

Java 18

- 408: Simple Web Server
- 413: Code Snippets in Java API Documentation
- 417: Vector API (Third Incubator)
- 419: Foreign Function & Memory API (Second Incubator)
- 420: Pattern Matching for switch (Second Preview)
- **421: Deprecate Finalization for Removal**

```
public class CleaningExample implements AutoCloseable {

    private static final Cleaner cleaner = <cleaner>;

    private final State state;
    private final Cleaner.Cleanable cleanable;

    public CleaningExample() {
        this.state = new State(...);
        this.cleanable = cleaner.register(this, state);
    }

    public void close() {
        cleanable.clean();
    }

}
```

```
static class State implements Runnable {
    State(...) {
        // Initialisierung.
    }

    public void run() {
        // Aufräumen.
    }
}
```

Java 19

- 405: Record Patterns (Preview)
- ~~422: Linux/RISC-V Port~~
- ~~424: Foreign Function & Memory API (Preview)~~
- 425: Virtual Threads (Preview)
- ~~426: Vector API (Fourth Incubator)~~
- ~~427: Pattern Matching for switch (Third Preview)~~
- 428: Structured Concurrency (Incubator)

Java 19

- 405: Record Patterns (Preview)
- 425: Virtual Threads (Preview)
- 428: Structured Concurrency (Incubator)

Java 19

- **405: Record Patterns (Preview)**
- 425: Virtual Threads (Preview)
- 428: Structured Concurrency (Incubator)

```
// Vorher
if (o instanceof String) {
    String s = (String) o;
    ℰ(s)
}
```

```
// Nachher
if (o instanceof String s) {
    ℰ(s)
}
```

record Point(int *x*, int *y*) { }

```
// Vorher
static void printSum(Object o) {
    if (o instanceof Point p) {
        int x = p.x();
        int y = p.y();
        System.out.println(x + y);
    }
}
```

```
// Nachher
static void printSum(Object o) {
    if (o instanceof Point(int x, int y)) {
        System.out.println(x + y);
    }
}
```

```
record Point(int x, int y) {}

enum Color {RED, GREEN, BLUE}

record ColoredPoint(Point p, Color c) {}

record Rectangle(ColoredPoint upperLeft, ColoredPoint lowerRight) {}


static void printUpperLeftColoredPoint(Rectangle r) {
    if (r instanceof Rectangle(ColoredPoint ul, ColoredPoint lr)) {
        System.out.println(ul.c());
    }
}
```

```
record Point(int x, int y) {}

enum Color {RED, GREEN, BLUE}

record ColoredPoint(Point p, Color c) {}

record Rectangle(ColoredPoint upperLeft, ColoredPoint lowerRight) {}


static void printColorOfUpperLeftPoint(Rectangle r) {
    if (r instanceof Rectangle(ColoredPoint(Point p, Color c), ColoredPoint lr)) {
        System.out.println(c);
    }
}
```

```
record Point(int x, int y) {}

enum Color {RED, GREEN, BLUE}

record ColoredPoint(Point p, Color c) {}

record Rectangle(ColoredPoint upperLeft, ColoredPoint lowerRight) {}


static void printXCoordOfUpperLeftPointWithPatterns(Rectangle r) {
    if (r instanceof Rectangle(ColoredPoint(Point(var x, var y), var c), var lr)) {
        System.out.println("Upper-left corner: " + x);
    }
}
```

Java 19

- 405: Record Patterns (Preview)
- **425: Virtual Threads (Preview)**
- 428: Structured Concurrency (Incubator)

```
Thread.startVirtualThread(
    () –> {
        System.out.println("Hello World");
    }
);
```

Java 19

- 405: Record Patterns (Preview)
- 425: Virtual Threads (Preview)
- **428: Structured Concurrency (Incubator)**

```
Response handle() throws IOException {
    String theUser = findUser();
    int theOrder = fetchOrder();
    return new Response(theUser, theOrder);
}
```

```
Response handle() throws ExecutionException, InterruptedException {
    Future<String>  user  = es.submit(() -> findUser());
    Future<Integer> order = es.submit(() -> fetchOrder());
    String theUser  = user.get();
    int theOrder = order.get();
    return new Response(theUser, theOrder);
}
```

```
Response handle() throws ExecutionException, InterruptedException {
    Future<String>  user  = es.submit(() –> findUser());                  // Ausnahme
    Future<Integer> order = es.submit(() –> fetchOrder());
    String theUser  = user.get();   // Join findUser
    int theOrder = order.get();  // Join fetchOrder
    return new Response(theUser, theOrder);
}
```

```
Response handle() throws ExecutionException, InterruptedException {
    Future<String>  user  = es.submit(() -> findUser());              // Braucht lang.
    Future<Integer> order = es.submit(() -> fetchOrder());            // Versagt!
    String theUser  = user.get();   // Join findUser
    int theOrder = order.get();  // Join fetchOrder
    return new Response(theUser, theOrder);
}
```

```
Response handle() throws ExecutionException, InterruptedException {
    try (var scope = new StructuredTaskScope.ShutdownOnFailure()) {
        Future<String> user = scope.fork(() -> findUser());
        Future<Integer> order = scope.fork(() -> fetchOrder());
        scope.join();
        scope.throwIfFailed();
        return new Response(user.resultNow(), order.resultNow());
    }
}
```

Amber

- Sealed classes
- Pattern matching in switches
- Pattern matching for records and arrays

Valhalla

- Preparatory changes
  - JEP 181: Nest-Based Access Control (11)
  - JEP 309: Dynamic Class-File Constants (11)
  - JEP 371: Hidden Classes (15)
  - JEP 390: Warnings for Value-Based Classes (16)
  - Better-defined JVM class file validation (draft)
- Value objects
  - Value Objects (Preview) (submitted draft)
  - JEP 401: Primitive Classes (Preview) (candidate)
  - JEP 402: Classes for the Basic Primitives (Preview) (candidate)
- Enhanced generics
  - Universal Generics (Preview) (submitted draft)
  - Parametric JVM (no draft yet)

Collection<int>

Map<int, double>

Map<Integer, Double>

Map<Integer, double>

Panama

- Foreign-Memory Access API
- Foreign Linker API
- The Vector API

Loom

- Fibers
- Continuations
- Tail-Call Eliminierung

Leyden

- Native Images
- Schnellere Startzeiten

Vielen Dank!