Hilfreiche Neuerungen für Softwareentwickler in Java7 und Java8 abseits von Lambdas

David Burkhart

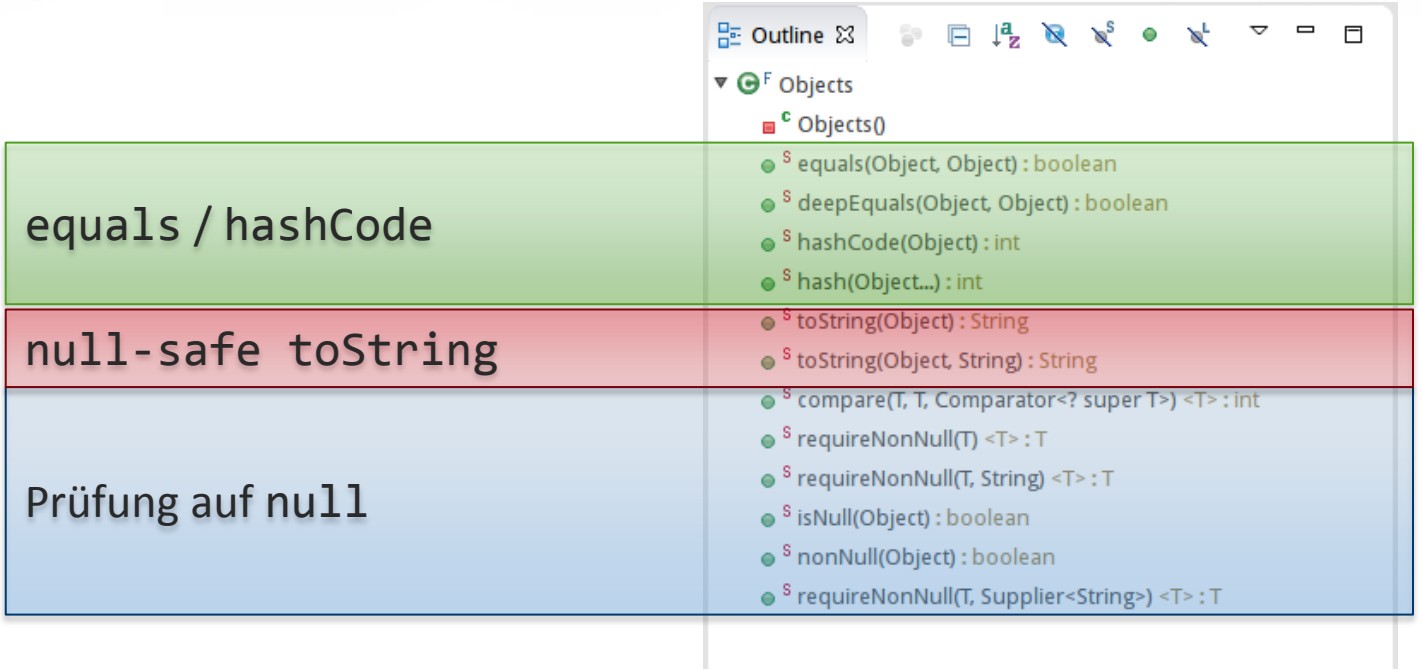
Simon Wagner

Entwicklertag Karlsruhe 2015

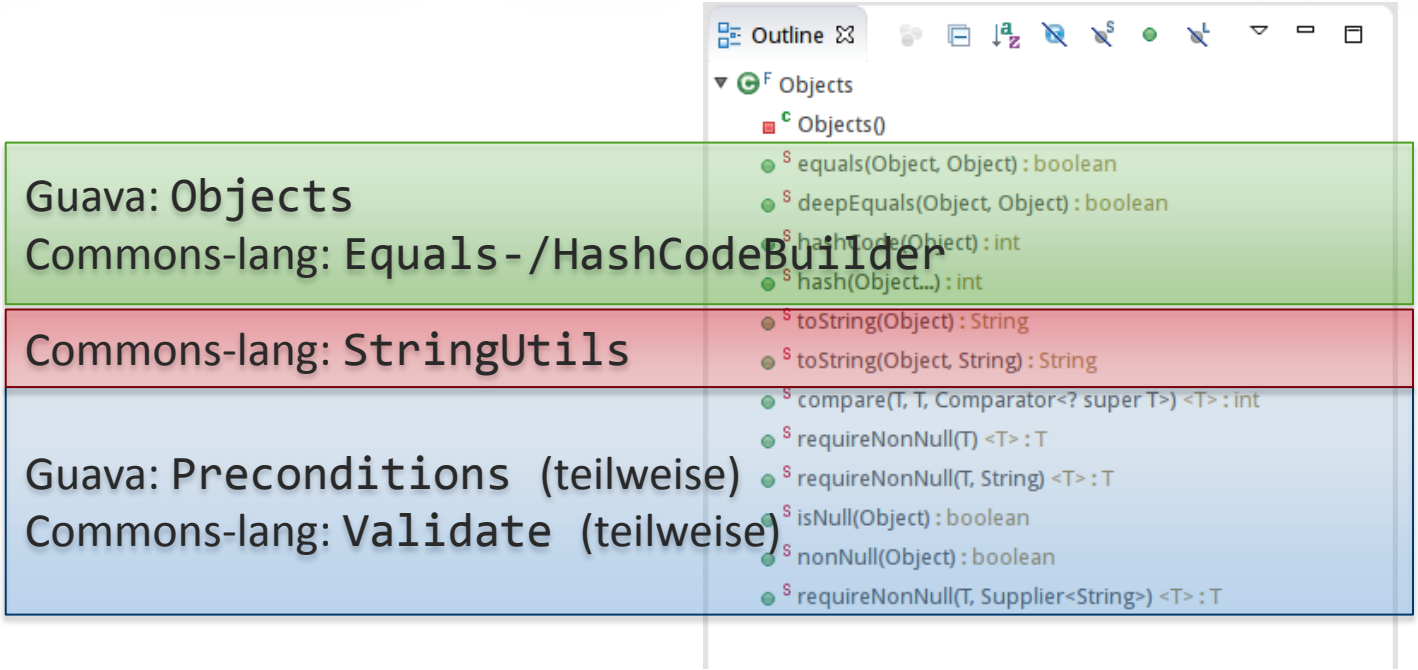
Kleine Sprachverbesserungen

Neue Klassen, neue Sprach-Features…

java.util.Objects



java.util.Objects



Guava: Objects
Commons-lang: Equals-/HashCodeBuilder

Commons-lang: StringUtils

Guava: Preconditions (teilweise)
Commons-lang: Validate (teilweise)

String Joining – Apache Commons

```
import org.apache.commons.lang3.StringUtils;
List<String> cityList = Arrays.asList("KA", "Köln", "Berlin");

String variante1 = StringUtils.join("Karlsruhe", "Köln", "Berlin");
String variante2 = StringUtils.join(cityList, " | ");

assertThat(variante1, is(equalTo("KarlsruheKölnBerlin")));
assertThat(variante2, is(equalTo("Karlsruhe | Köln | Berlin")));
```


String Joining – Google Guava

```
import com.google.common.base.Joiner;
List<String> cityList = Arrays.asList("KA", "Köln", "Berlin");

String joined = Joiner.on(", ").join(cityList);

assertThat(joined, is(equalTo("Karlsruhe, Köln, Berlin")));
```


String Joining – Java8: String.join()

```
List<String> cityList = Arrays.asList("KA", "Köln", "Berlin");

String variante1 = String.join(", ", cityList);
String variante2 = String.join(", ", "Karlsruhe", "Köln", Berlin");

assertThat(variante1, is(equalTo("Karlsruhe, Köln, Berlin")));
assertThat(variante2, is(equalTo("Karlsruhe, Köln, Berlin")));
```


String Joining – Java8: Collectors.joining()

```
String variante1 =
    cityList.stream().collect(Collectors.joining());
String variante2 =
    cityList.stream().collect(Collectors.joining(" | "));
String variante3 =
    cityList.stream().collect(Collectors.joining(" | ", "[", "]"));

assertThat(variante1, is(equalTo("KarlsruheKölnBerlin")));
assertThat(variante2, is(equalTo("Karlsruhe | Köln | Berlin")));
assertThat(variante3, is(equalTo("[Karlsruhe | Köln | Berlin]")));
```

String Joining – Java8: StringJoiner

```
StringJoiner stringJoiner = new StringJoiner(", ", "", ".");
// Alternative ohne Prefix und Suffix: new StringJoiner(", ");
for (String city : cityList) {
    stringJoiner.add(city);
}

assertThat(stringJoiner.toString(),
                is(equalTo("Karlsruhe, Köln, Berlin.")));
```


Klassischer Umgang mit schließbaren Resourcen

```
try {
    InputStream input = null;
    try {
        input = new FileInputStream("file.txt");
        // TODO read from input
    } finally {
        if (input != null) input.close();
    }
} catch (IOException e) {
    // TODO handle e
}
```


Probiers mal mit … try with

```
try (InputStream input = new FileInputStream("file.txt")) {
    // TODO read from input
} catch (IOException e) {
    // TODO handle e

    // TODO handle Exception[] suppressed = e.getSuppressed();
    // contains Exception thrown by close() of input
}
```


Probiers mal mit … try with

```
try (InputStream input = new FileInputStream("file.txt")) {
    // TODO read from inpu
} catch (IOException e) {
    // TODO handle e

    // TODO handle Excepti
    // contains Exception
}
```

Try-with funktioniert mit:

```
interface AutoCloseable {
    void close() throws Exception;
}
```

Klassisches Exception-handling

```
try {
    dangerousCode();
} catch (NumberFormatException e) {
    handleFormatException(e);
} catch (DateTimeParseException e) {
    handleFormatException(e);
} catch (Exception e) {
    handleTechnicalException(e);
}
```


Multicatch – eine Kurzschreibweise

```
try {
    dangerousCode();
} catch (NumberFormatException | DateTimeParseException e) {
    handleFormatException(e);
} catch (Exception e) {
    handleTechnicalException(e);
}
```


Base64 – Apache Commons

```
import org.apache.commons.codec.binary.Base64;

String encoded =
    Base64.encodeBase64String("user:password".getBytes("UTF8"));
assertThat(encoded, is(equalTo("dXNlcjpwYXNzd29yZA==")));

byte[] decoded = Base64.decodeBase64("dXNlcjpwYXNzd29yZA==");
assertThat(new String(decoded, "UTF8"), is(equalTo("user:password")));
```


Base64 – Google Guava

```
import com.google.common.io.BaseEncoding;

String encoded =
    BaseEncoding.base64().encode("user:password".getBytes("UTF8"));
assertThat(encoded, is(equalTo("dXNlcjpwYXNzd29yZA==")));

byte[] decoded = BaseEncoding.base64().decode("dXNlcjpwYXNzd29yZA==");
assertThat(new String(decoded, "UTF8"), is(equalTo("user:password")));
```


Base64 – Java8

```
import java.util.Base64;

String encoded = Base64.getEncoder()
        .encodeToString("user:password".getBytes("UTF8"));
assertThat(encoded, is(equalTo("dXNlcjpwYXNzd29yZA==")));

byte[] decoded = Base64.getDecoder().decode("dXNlcjpwYXNzd29yZA==");
assertThat(new String(decoded, "UTF8"), is(equalTo("user:password")));
```


Base64 – Java8

- Weitere En-/Decoder
- `Base64.getMimeEncoder();`
- `Base64.getMimeDecoder();`
- `Base64.getUrlEncoder();`
- `Base64.getUrlDecoder();`

- Hinweis: Es gibt eine nicht unterstützte Base64 Implementierung in Java. Vorsicht, in Java 9 nicht mehr möglich!

NIO.2

Neue File-API…

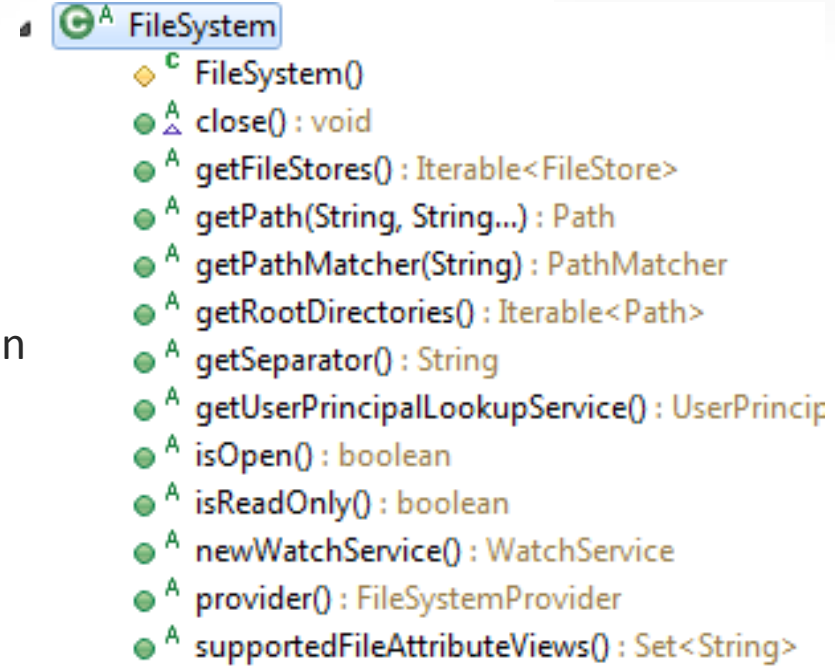
NIO.2 – Path, Paths

- NIO.2
  - Mit Java7 eingeführt
  - Nachfolger der NIO („new I/O“) aus Java 1.4

- Path ist ein Interface
  - Dateisystemabhängige Implementierungen
  - Keine Verbindung zu einer konkreten Datei
- Hilfsklasse Paths

NIO.2 – FileSystem

- Abstakte Klasse FileSystem
  - Erzeugung von Pfad-Objekten
  - Ermittlung Wurzelverzeichnis
- Dateisystemabhängige Implementierungen
  - LinuxFileSystem
  - WindowsFileSystem
  - ZipFileSystem



NIO.2 – Files

- Hilfsklasse mit Methoden zum Umgang mit
  - Dateien und Verzeichnissen
  - Symbolischen Links
  - Dateiattribute

NIO.2 – Files

- Dateien lesen und schreiben
- Rekursives Ablaufen von Verzeichnissen

java.time

DateTime-API Versuch #3…

java.time

- Immutable, Threadsafe (auch Formatter!)
- Brücken zur alten Welt

```
java.util.Date: from(Instant), toInstant()
java.sql.Date: from(Instant), toInstant(), valueOf(LocalDate), toLocalDate()
java.util.Calendar: toInstant()
java.sql.Timestamp: valueOf(LocalDateTime), toLocalDateTime()
java.sql.Timestamp: from(Instant), toInstant()
```

- Warum nicht Joda-Time?
  http://blog.joda.org/2009/11/why-jsr-310-isn-joda-time_4941.html
- API-Doku:
  https://docs.oracle.com/javase/8/docs/api/java/time/package-summary.html

java.time

Maschine

- Instant

java.util.Date

Menschenlesbar

- LocalDate
- LocalTime
- LocalDateTime

- Enum: DayOfWeek
- Enum: Month
- MonthDay
- Year
- YearMonth

Mit Zeitzonen

- ZonedDateTime
- OffsetDateTime
- OffsetTime

java.util.Calendar

java.time

Maschine	Menschenlesbar	Mit Zeitzonen
• Insta		e me

Where possible, it is recommended to use a simpler class without a time-zone. The widespread **use of time-zones** tends to **add considerable complexity** to an application.

java.util.Date

java.util.Calendar

java.time

Duration

- Sekunden
- Nanosekunden

Period

- Jahre
- Monate
- Tage

Weiteres

- Optional (in Verbindung mit Lambdas)
- Java FX 8
- Concurrency Verbesserungen (CompletableFuture, ...)
- Nashorn Javascript Engine
- switch über Strings

Links

http://www.oracle.com/technetwork/java/javase/downloads

https://github.com/google/guava

http://commons.apache.org