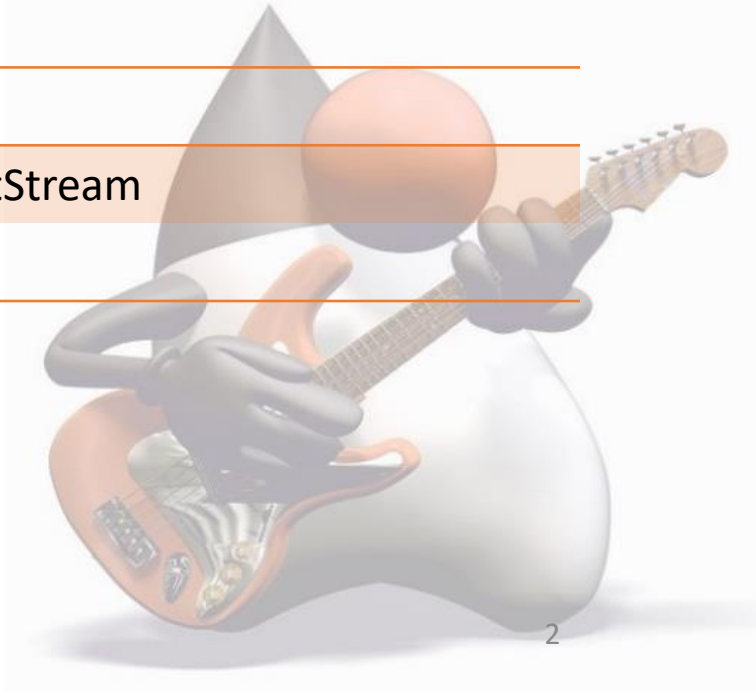
Java Fundamentals

Java IO

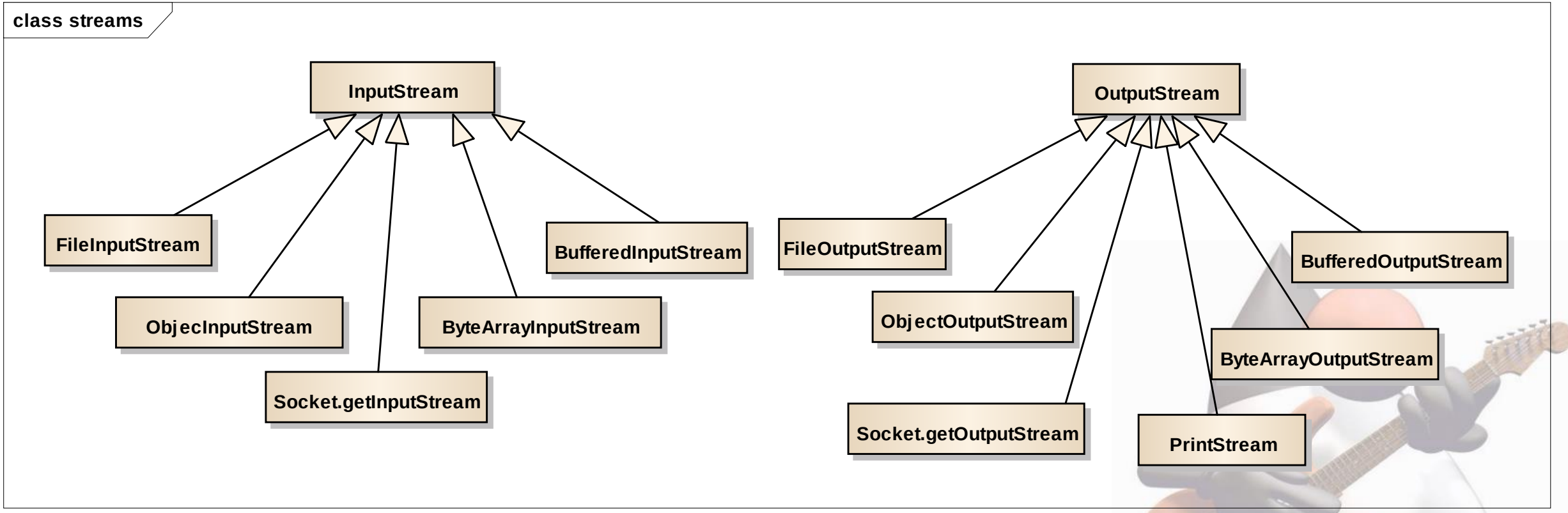
Streams

- Package java.io
- Lesen und Schreiben von Binär- und Characterdaten
- Quellen und Senken können sein:
  Dateien, Netzwerk, Speicher, Datenbank, usf.

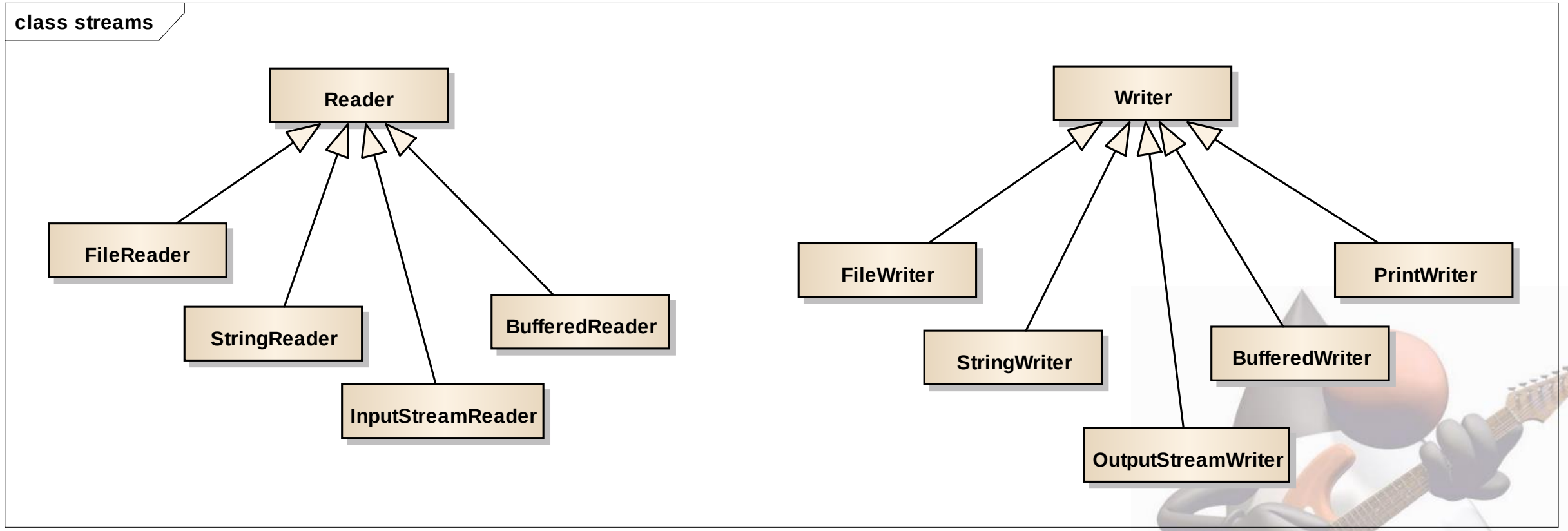
	Read	**Write**
Bytes	InputStream	OutputStream
Characters	Reader	Writer



Binär Streams

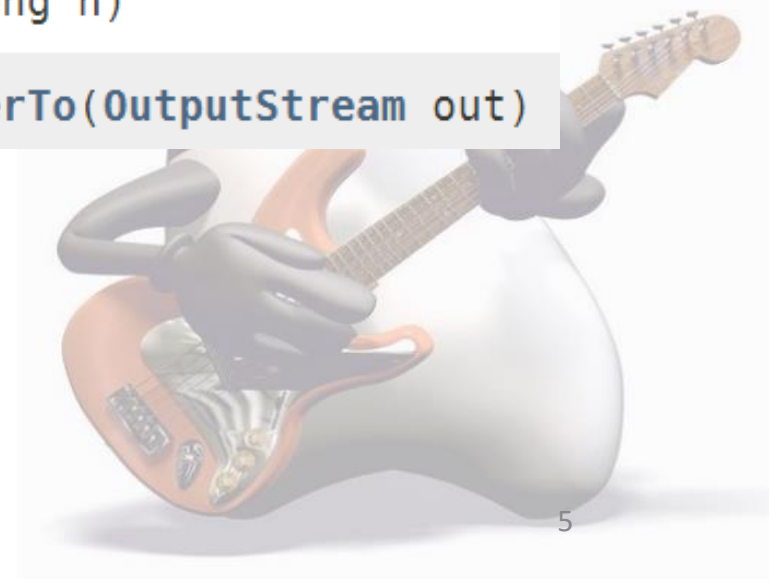


Character Streams



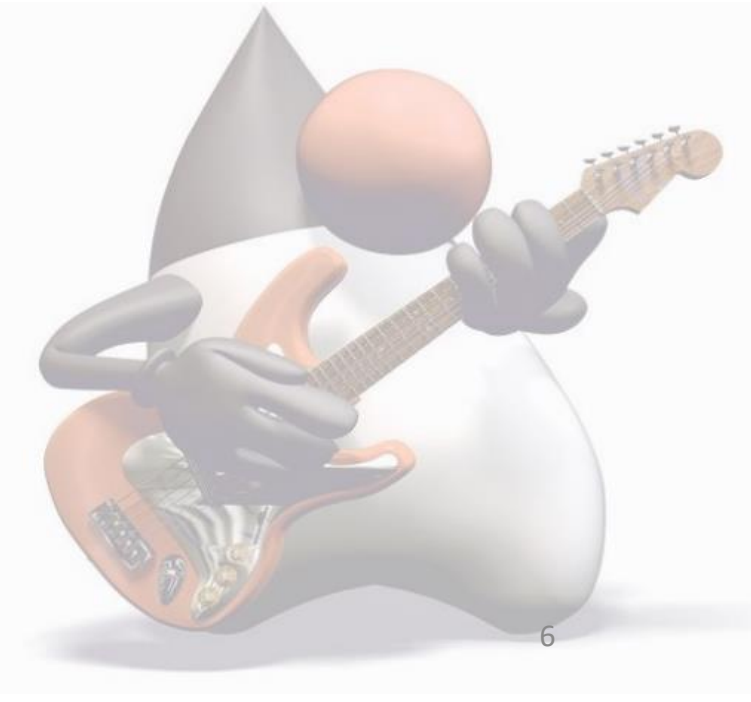
InputStream

int	available()
void	close()
void	mark(int readlimit)
boolean	markSupported()
static InputStream	nullInputStream()
abstract int	read()
int	read(byte[] b)
int	read(byte[] b, int off, int len)
byte[]	readAllBytes()
int	readNBytes(byte[] b, int off, int len)
byte[]	readNBytes(int len)
void	reset()
long	skip(long n)
long	transferTo(OutputStream out)



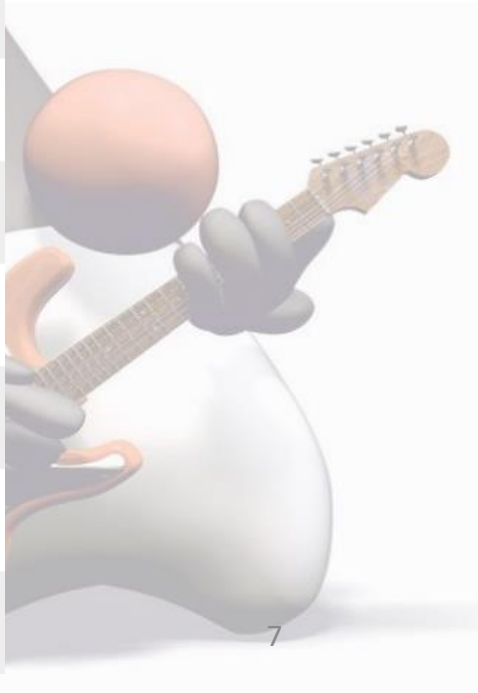
OutputStream

void	close()
void	flush()
static OutputStream	nullOutputStream()
void	write(byte[] b)
void	write(byte[] b, int off, int len)
abstract void	write(int b)



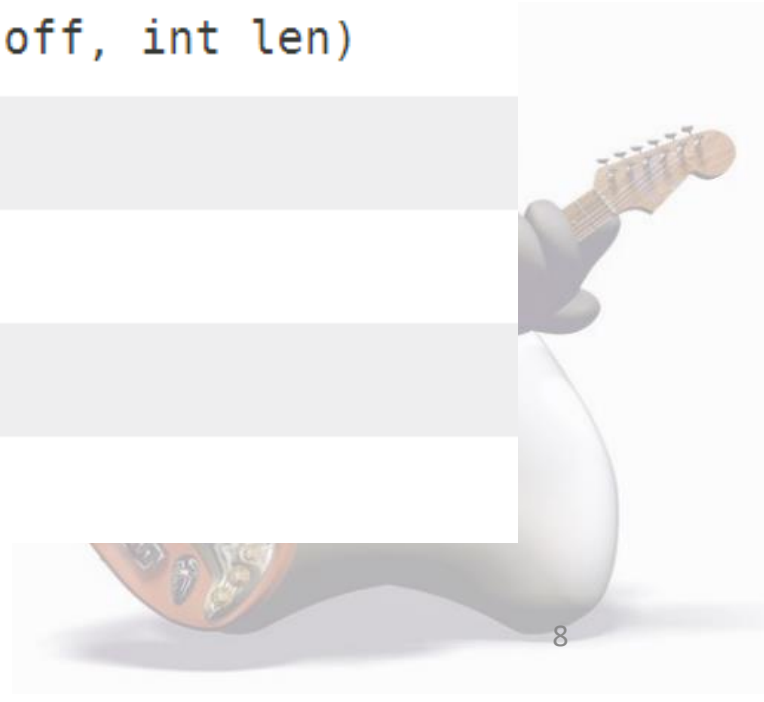
Reader

abstract void	close()
void	mark(int readAheadLimit)
boolean	markSupported()
static Reader	nullReader()
int	read()
int	read(char[] cbuf)
abstract int	read(char[] cbuf, int off, int len)
int	read(CharBuffer target)
boolean	ready()
void	reset()
long	skip(long n)
long	transferTo(Writer out)



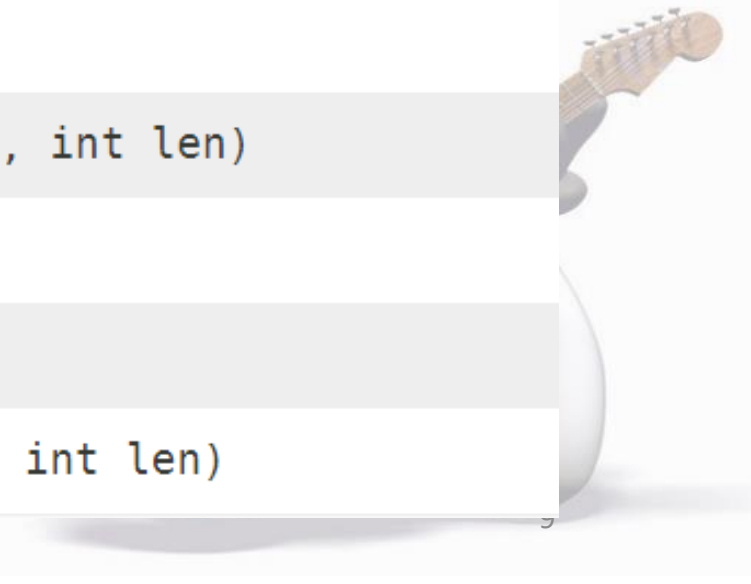
BufferedReader

Stream<String>	lines()
void	mark(int readAheadLimit)
boolean	markSupported()
int	read()
int	read(char[] cbuf, int off, int len)
String	readLine()
boolean	ready()
void	reset()
long	skip(long n)



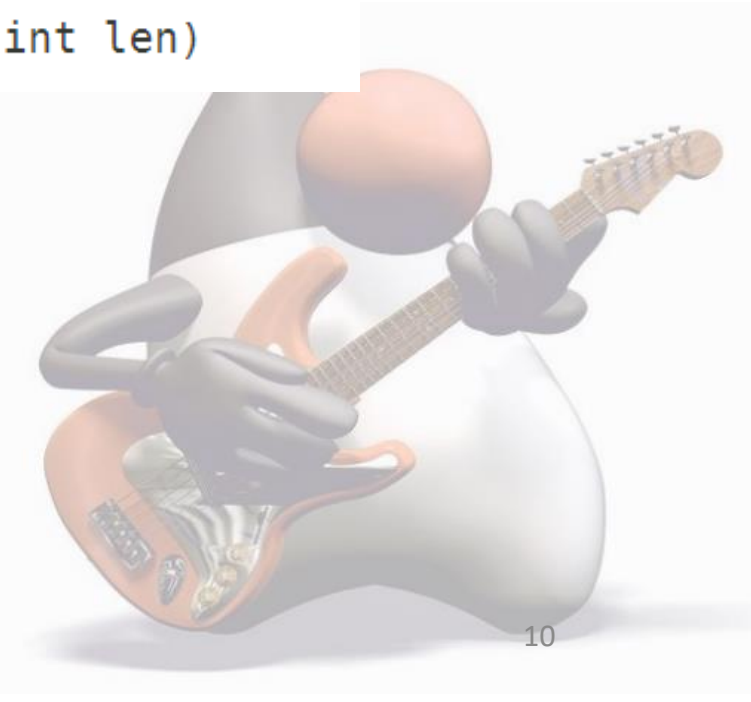
Writer

Writer	append(char c)
Writer	append(CharSequence csq)
Writer	append(CharSequence csq, int start, int end)
abstract void	close()
abstract void	flush()
static Writer	nullWriter()
void	write(char[] cbuf)
abstract void	write(char[] cbuf, int off, int len)
void	write(int c)
void	write(String str)
void	write(String str, int off, int len)



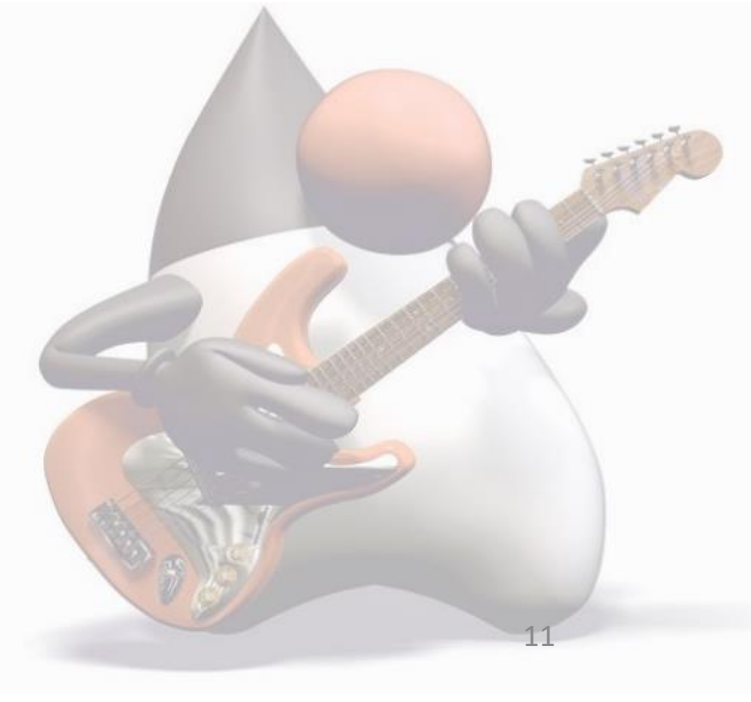
BufferedWriter

void	flush()
void	newLine()
void	write(char[] cbuf, int off, int len)
void	write(int c)
void	write(String s, int off, int len)

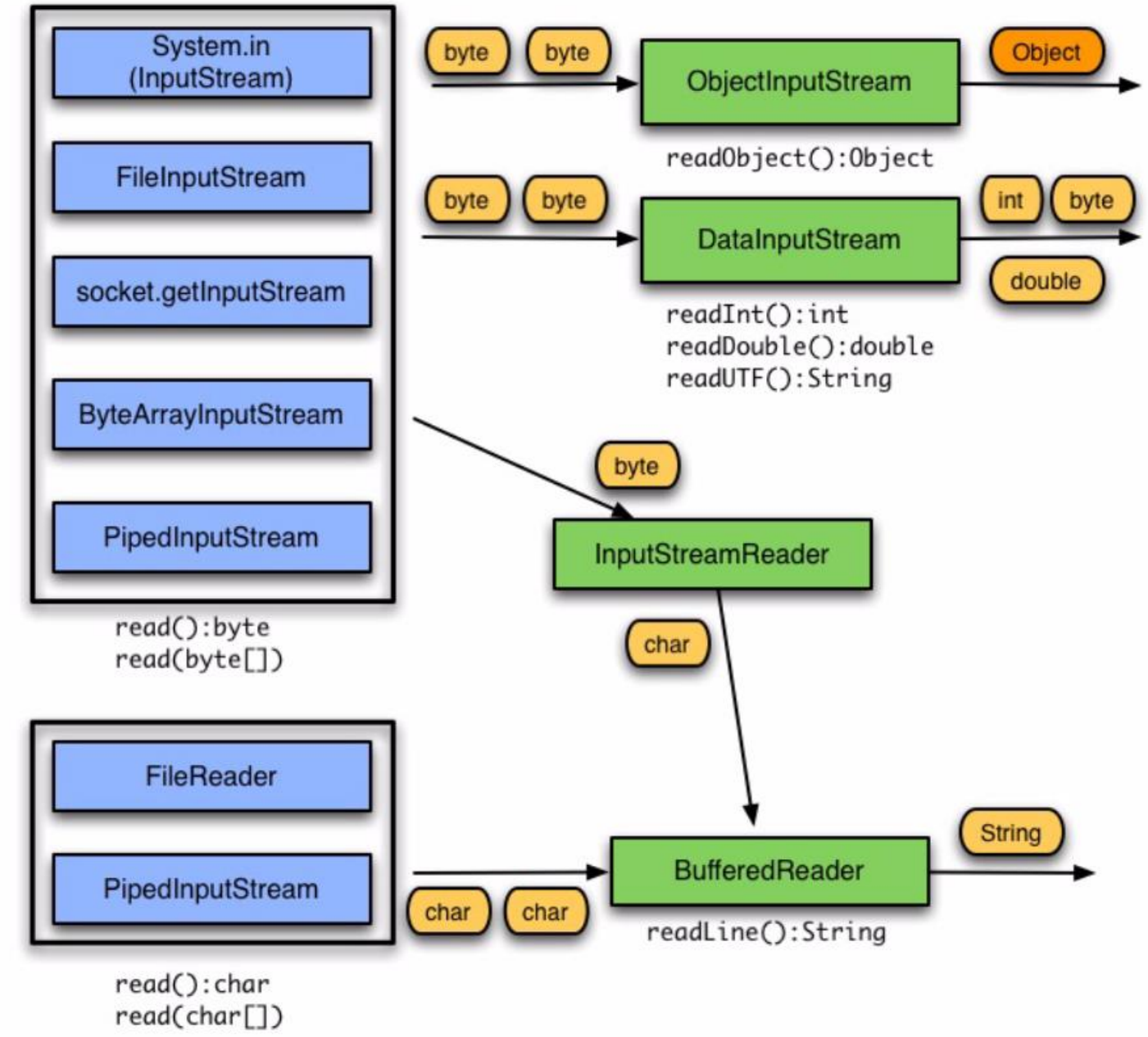


Schreiben in Textdatei

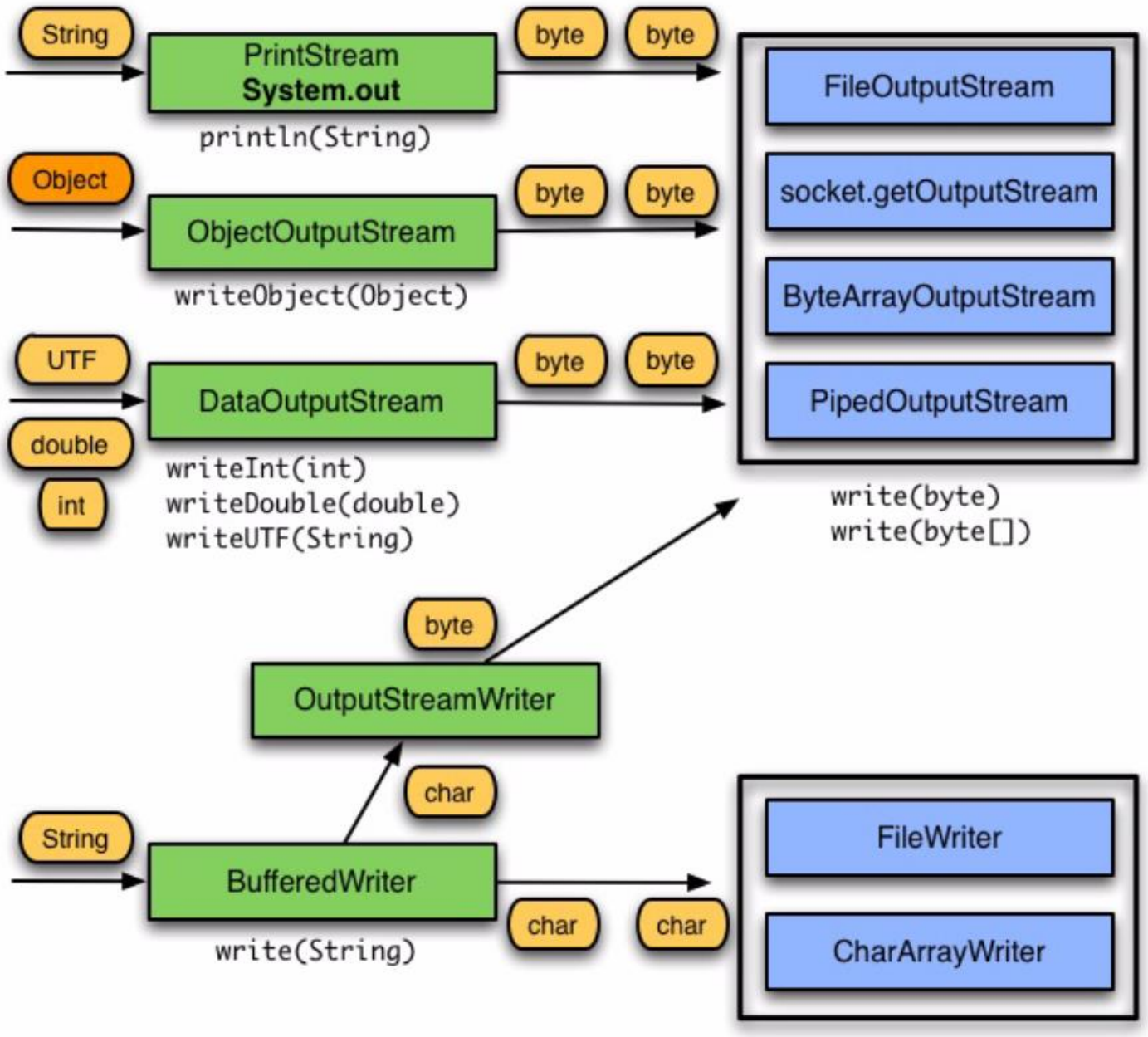
```
try (BufferedReader reader = new BufferedReader(new InputStreamReader(System.in))) {
        try (BufferedWriter writer = new BufferedWriter(new FileWriter("myfile.txt"))) {
                String line = reader.readLine();
                while (!line.equals("exit")) {
                        writer.write(line);
                        writer.newLine();
                        line = reader.readLine();
                }
        }
} catch (FileNotFoundException e) {
        e.printStackTrace();
} catch (IOException e) {
        e.printStackTrace();
}
```



Stream Chaining Input

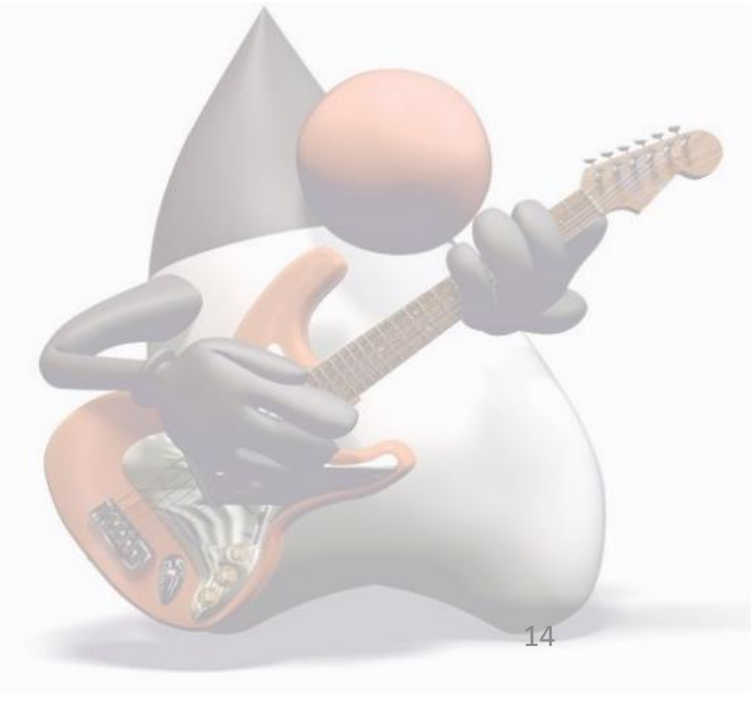


Stream Chaining Output



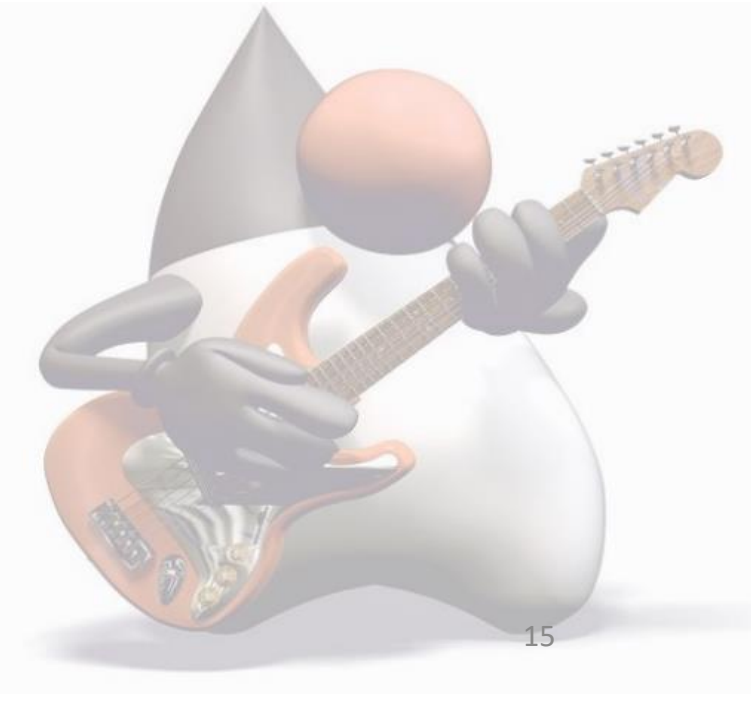
Schreiben in Binärdatei

```
public class Article implements Serializable{
        private long id;
        private String description;

        public Article(long id, String description) {
                this.id = id;
                this.description = description;
        }

        public long getId() {
                return id;
        }

        public void setId(long id) {
                this.id = id;
        }
…
}
```



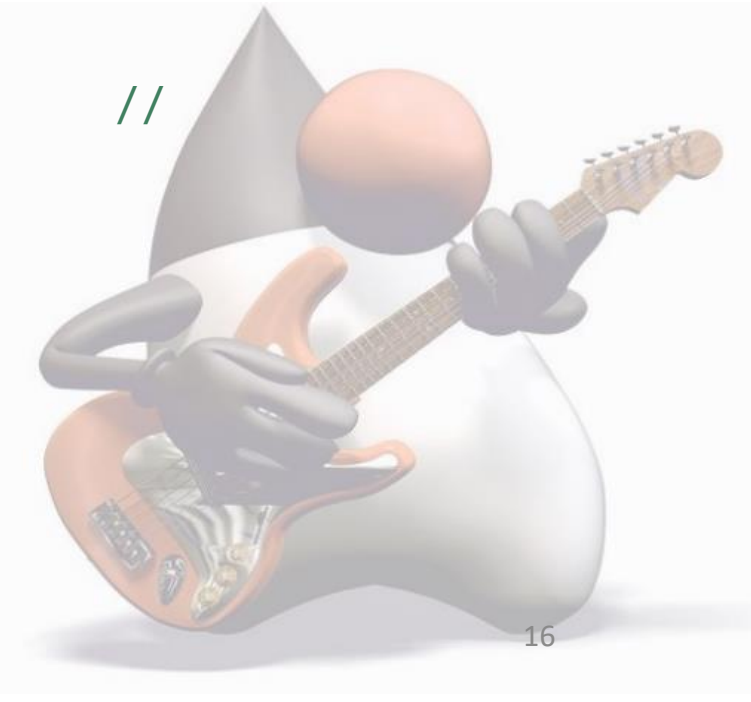
Schreiben in Binärdatei

```
public class ObjectWriter {
	public static void main(String[] args) {
		try (ObjectOutputStream ois =
				new ObjectOutputStream(new FileOutputStream("article.bin")))

			Article article = new Article(1, "T-Shirt");
			ois.writeObject(article);
		} catch (IOException e) {
			e.printStackTrace();
		}
	}
}
```



Lesen aus Binärdatei

```
public class ObjectReader {
	public static void main(String[] args) {
		try(ObjectInputStream ois =
				new ObjectInputStream(new FileInputStream("article.bin"))){
			Article article = (Article)ois.readObject();
			System.out.println(article);
		} catch (IOException e) {
			e.printStackTrace();
		} catch (ClassNotFoundException e) {
			e.printStackTrace();
		}
	}
}
```

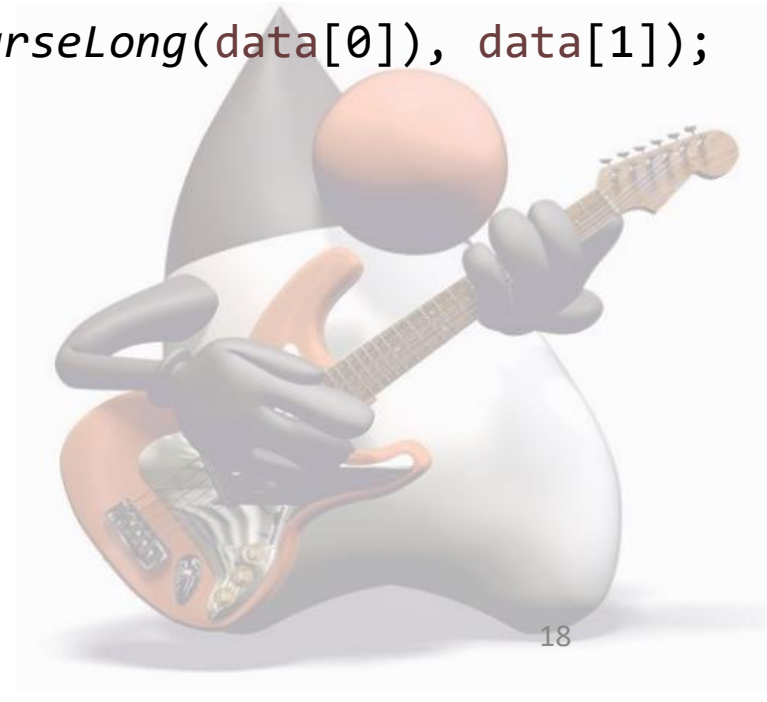


Schreiben csv Datei

```
public class CsvWriter {
	public static void main(String[] args) {
		Article a1 = new Article(2, "Trousers");
		Article a2 = new Article(3, "Scarf");
		List<Article> articleList = Arrays.asList(a1, a2);
		try(PrintWriter writer = new PrintWriter(new FileWriter("articles.csv"))){
			for (Article article:articleList) {
			  writer.println(article.getId() + "," + article.getDescription());
			}
		} catch (IOException e) {
			e.printStackTrace();
		}
	}
}
```

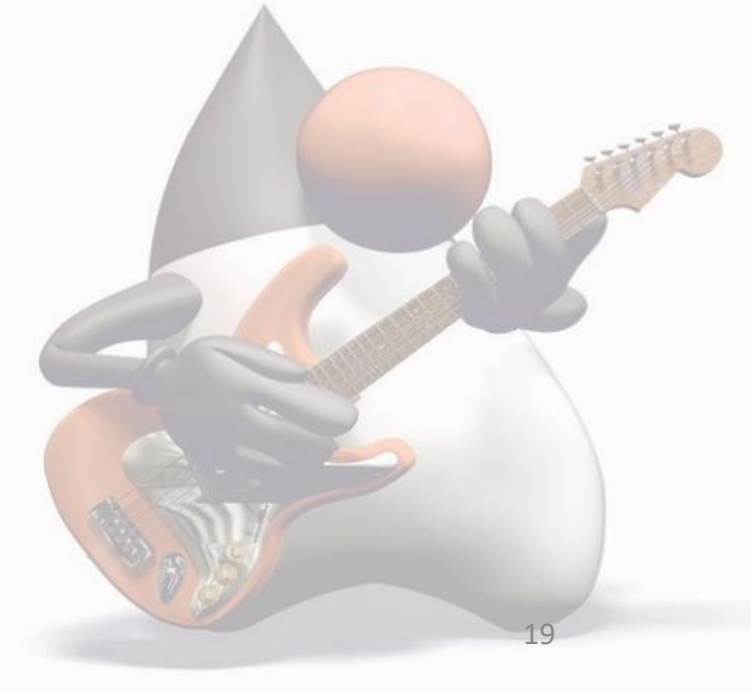
Lesen csv Datei

```
public class CsvReader {
	public static void main(String[] args) {
		try (BufferedReader reader =
				Files.newBufferedReader(Paths.get("articles.csv"))) {
			String line = reader.readLine();
			while (line != null) {
				String[] data = line.split(",");
				Article article = new Article(Long.parseLong(data[0]), data[1]);
				System.out.println(article);
				line = reader.readLine();
			}
		} catch (IOException e) {
			e.printStackTrace();
		}
	}
}
```



Übung

- Schreiben Sie eine Klasse AccountWriter, die einen Account oder eine List<Account> in eine Textdatei schreiben kann.
- Schreiben Sie eine Klasse AccountReader, die einen oder mehrere Accounts aus einer Textdatei lesen kann.



Java Fundamentals

Java IO