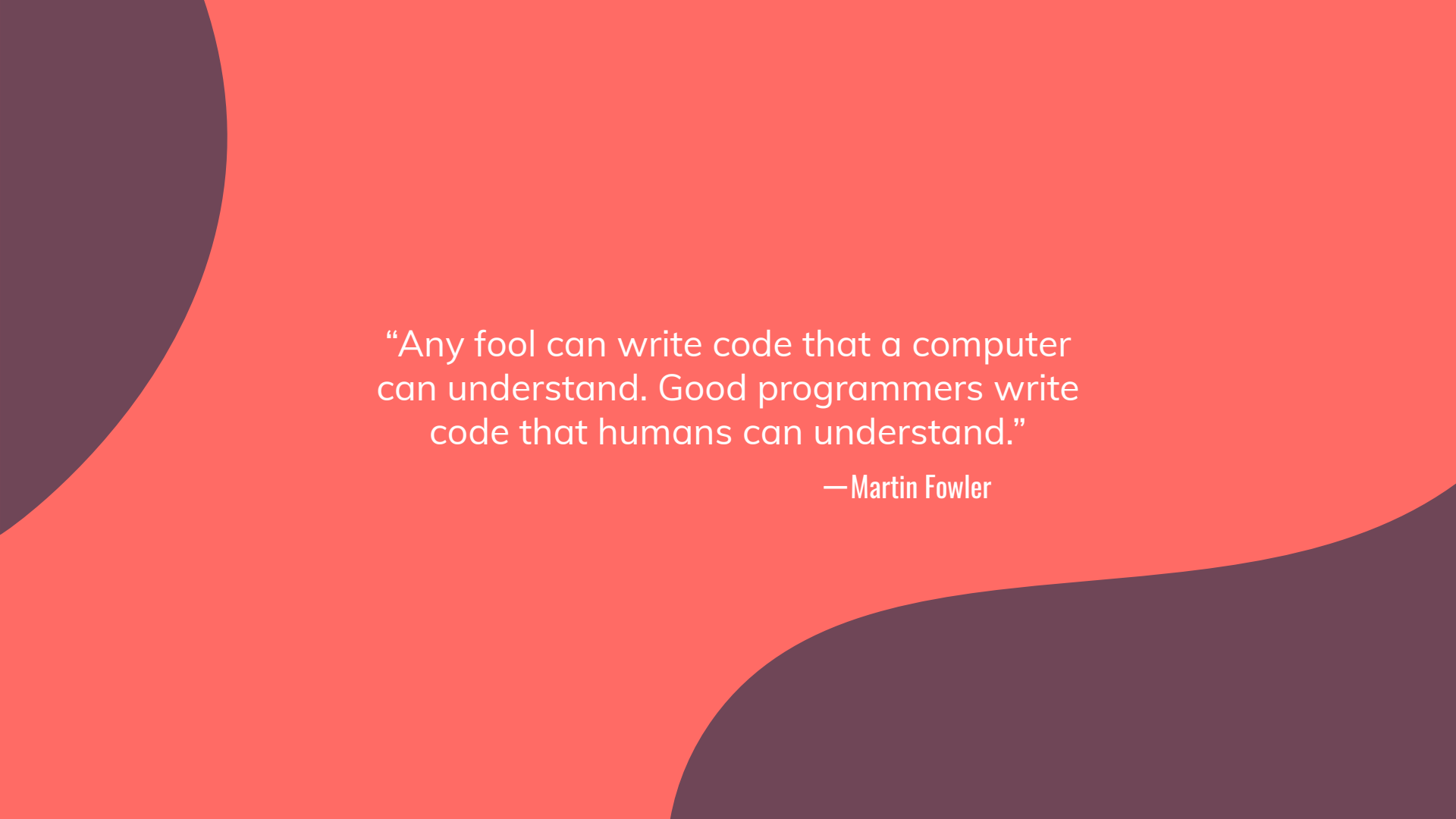


Coding 1

- Einführung in den Workshop
- Grundlegende Elemente der Programmierung



“Any fool can write code that a computer
can understand. Good programmers write
code that humans can understand.”

—Martin Fowler

EINFÜHRUNG

1

VARIABLEN

2

BEDINGUNGEN

3

SCHLEIFEN

4

INHALT



Michael Bykovski // bykovski.de

- _ 14 Jahre Programmierer
- _ //SEIBERT/MEDIA, Scholz & Volkmer, AOE...
- _ Full-Stack Senior Software Engineer and Architect
- _ Master und Lehrauftrag an der HSRM

The background features a large, abstract composition of organic, flowing shapes. A vibrant red shape dominates the left and bottom portions of the frame. A teal-colored shape occupies the top right and extends down the right side. A white, angular shape is positioned in the upper left, partially overlapping the red and teal areas. The text 'ERWARTUNGEN?' is centered within this white shape.

ERWARTUNGEN?

TAG 1

- **Coding 1**
 - Einführung in den Workshop
 - Grundlegende Elemente der Programmierung
- **Web 1**
 - Einführung in die Webentwicklung
- **Web 2**
 - HTML, CSS, JS
- **Web Hands-On**
 - Eigene kleine Webseite entwickeln

TAG 2

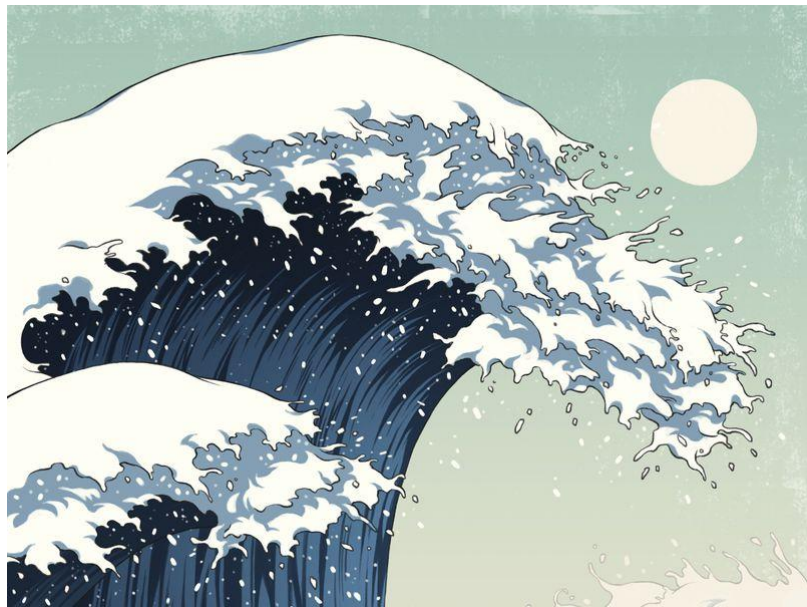
- Coding 2
 - Python
- Python for Business 1
 - Einlesen, Manipulieren, Anzeige von Daten (CSV, Excel)
- Python for Business 2
 - Einfache benutzerdefinierte Auswertung von Daten
- Python for Business Hands-On
 - Umsetzung der in Python for Business erlangten Kenntnisse

Alle Folien

<https://github.com/bykof/coding-workshop>



**“PROGRAMMIERER
WIRD MAN NUR
DURCH
PROGRAMMIEREN”**



LINES OF CODE

990 LINES

Kleine Bibliothek, die eine
Zustandsmaschine abbildet

396.053 LINES

Django Framework
Ein Web-Framework in
Python

60.000.000 LINES

Facebook



Code



Programm

Programmcode

```
>>> a = 2
```

```
>>> b = 3
```

```
>>> a + b
```

```
>>> 5
```



Maschinencode

```
0010010010001
```

```
0010010001000
```

```
0100101000010
```

Kompilieren

Code

```
>>> a = 2  
>>> b = 3  
>>> a + b  
5
```



Maschinen Code

```
0010010010  
0100100101  
0010101010  
1001010010
```

Interpretieren

Programm

```
>>> a = 2
>>> b = 3
>>> a + b
5
```

Back to Basics

The background features a minimalist design with large, organic shapes. A teal shape occupies the left and bottom-left areas. A white shape is at the top, containing the text. A large maroon shape dominates the right and bottom-right areas. A small red circle is partially visible on the left edge.



Variablen

Bedingungen

Schleifen

Variablen

```
x = 2  
y = 'Hallo Welt'
```

```
a = 2  
var b = 'a'  
var c: float = 2.4  
var d bool = true
```

Integer

a = 97

Im Speicher
01100001

Character

b = 'a'

Im Speicher
01100001

String

```
b = 'abc'
```

Im Speicher

01100001

01100010

01100011

Float

c = -2.5

Im Speicher

1100000000100000000000000000000000

Boolean

d = True

Im Speicher
1



Operatoren

+, -

Addition
Subtraktion

2+2

$x \circ y = z$
 $x, y, z \in M$

*, /, %

Multiplikati
on,
Division,
Rest

2*2

$x \circ y = z$
 $x, y, z \in M$

or, and,
not, &&,
||, !

Boolsche
Operationen

(a or b) and c
erzeugt bool

<, <=,
>,
>=,
!=,
==

Vergleichsop
erator

2 > 4
a == b

erzeugt bool

```
a = 2
b = 3
c = a + b // Ergebnis der Operation
d = c * 2
c = d / 5 // c wird überschrieben
```

c?

```
word = "te" + "st"
```

**Wie viele Variablen
kann ich in einem
Programm
definieren?**

a = 2

a = "abc"

a = -3.5

Bedingungen

```
if a == 2:  
    ...  
else:  
    ...
```

Operatoren

`+, -`

Addition
Subtraktion

`2+2`

$x \circ y = z$
 $x, y, z \in M$

`*, /, %`

Multiplikati
on,
Division,
Rest

`2*2`

$x \circ y = z$
 $x, y, z \in M$

`or, and,
not`

Boolsche
Operationen

`(a or b) and c`
erzeugt bool

`in`

Element in
Menge

`"2" in l`
erzeugt bool

`<, <=,
>,
>=,
!=,
==`

Vergleichsop
erator

`2 > 4`
`a == b`

erzeugt bool


```
a = 2  
b = 5
```

```
if :  
    ...  
else:  
    ...
```

```
a = "Hello World"  
b = "Hello WörlD"
```

```
if:  
    ...  
else:  
    ...
```

```
a = 2  
b = 2.0
```

```
if :  
    ...  
else:  
    ...
```

```
a = 1  
b = 2  
c = 3
```

```
if a > b:
```

```
...
```

```
elif b > c:
```

```
...
```

```
else:
```

```
...
```

Schleifen

```
for (i = 0; i < 0; i ++ ) {  
    ...  
}
```

2. Arten von Schleifen

- Für
 - Für jedes $e \in P$
- Solange
 - Solange $(b == \text{false})$ ist

$$M = \{1, 2, 3\}$$

$$e \in M$$

$$e = 1$$

$$e = ?$$

$$e = ?$$

$$M = \{1, \dots, 100\}$$

$$e \in M$$

$$e = 1$$

$$e = \dots$$

$$e = 100$$

**Problem:
Wir haben keine
Menge...**



i = 1

solange i > 1:
 führe aus...

wir definieren **i**,
solange **i** echt
größer 0...

Variablen

```
a = 2  
b = '2'  
c = 3.5  
d = 'Hello'
```

Bedingungen

```
a = 3  
b = 5  
  
if a < b:  
    ...  
else:  
    ...
```

Schleifen

```
für i in {1, 2, 3}:
```

```
...
```

```
i = True
```

```
solange i wahr ist:
```

```
...
```