

Exit Coding Game

# How to code in Python



Programmierung mit  
Python



# How to code Python

Weit verbreitete und einfach zu erlernende Programmiersprache.

Verwendet u.a. für:

- Webentwicklung
- Wissenschaft (u.a. Mathematik, Ingenieurwissenschaften)
- Data Science und Analytics
- uvm.



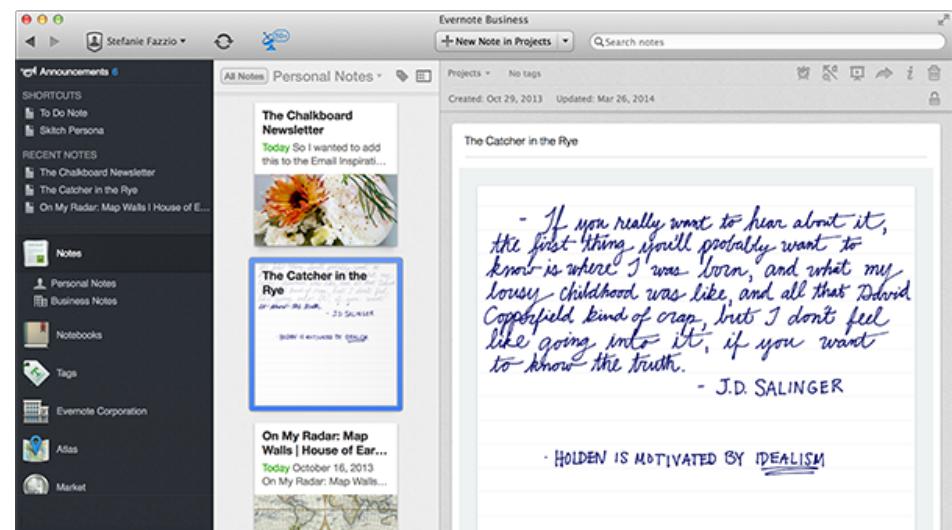
Bildquelle: <https://www.informatik-aktuell.de/betrieb/kuenstliche-intelligenz/data-science-buzzword-mit-gehaltvollem-kern.html>



## How to code

# Anwendungen für Texterkennung

- Digitalisierung handschriftlicher Notizen, um diese durchsuchbar und weiterverarbeitbar zu machen
- Digitalisierung von Formularen und Anträgen
- Digitalisierung und automatisierte Auswertung von Kassenzetteln als Basis für "Machine Learning"
- Online Überweisungen – App macht Screenshot der Rechnung
- uvm.



How to code

# Ausblick automatisierte Analyse von Bilddaten



Quelle: <https://netzpolitik.org/2019/deutsche-grossflughafen-gesichtserkennung-jetzt-auch-fuer-kinder/>

22.03.21

8



## How to code

# Arduino (C) vs. Python

**C:**

```
int LED_PIN = 12;

void setup() {
    pinMode(LED_PIN, OUTPUT);
}

void loop() {
    digitalWrite(LED_PIN, HIGH);
    delay(1000);
    digitalWrite(LED_PIN, LOW);
    delay(1000);
}
```

**Python:**

Definiert eine Funktion main – diese wird zum Start des Programms aufgerufen

```
def main():
    print("Hello world!")

if __name__ == "__main__":
    main()
```

Alle eingerückten (Tab-Taste) Anweisungen nach der Definition gehören zur Funktion, es gibt keine geschweiften Klammern!

Sorgt dafür dass die Funktion main() beim Start des Programms aufgerufen wird – kann man ansonsten ignorieren!



## How To code

# Back in time – Computer kann man auch ohne Fenster bedienen



Bildquelle: <https://www.worthpoint.com/worthopedia/sanyo-crt-30-monochrome-monitor-xt-1811071050>

- In den frühen 80er Jahren haben Computer nur getippte Befehle verstanden – Die Bedienung erfolgte über eine "Kommandozeile"
- Windows kam erst 1985 auf den Markt
- In der Softwareentwicklung und Systemadministration haben Kommandozeilen teilweise überlebt...

## How To code

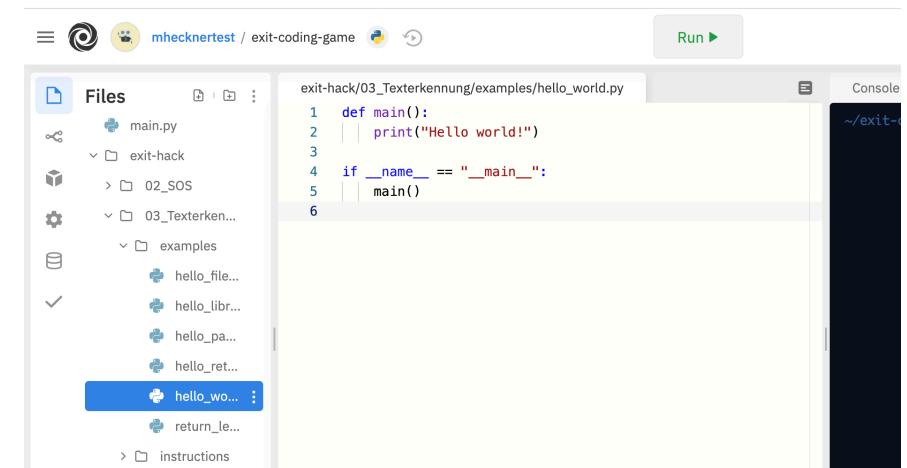
# Von der Arduino IDE zu Repl



```
demo_led_blink | Arduino 1.8.13
demo_led_blink
int LED_PIN = 12;

void setup() {
  pinMode(LED_PIN, OUTPUT);
}

void loop() {
  digitalWrite(LED_PIN, HIGH);
  delay(100);
  digitalWrite(LED_PIN, LOW);
  delay(100);
}
```



```
mheckner-test / exit-coding-game
exit-hack/03_Texterkennung/examples/hello_world.py
1 def main():
2     print("Hello world!")
3
4 if __name__ == "__main__":
5     main()
6
```

## Level 1 – Arduino IDE

22.03.21

11

## Level 2 – Repl: IDE für Python und Kommandozeile im Browser



## How to code

# Repl – Code Editor und Kommandozeile über den Browser in der Cloud

The screenshot shows a web-based development environment. On the left is a "File Explorer" (labeled "Datei-Explorer (wie in Windows)"), which lists files and folders. A blue box highlights the "main.py" file in the "exit-hack/03\_Texterkennung/examples/" directory. In the center is a "Text-Editor" (labeled "Text-Editor (hier schreibt man den Code)"), displaying the Python code for "hello\_world.py". The code prints "Hello world!" when run. On the right is a "Kommandozeile" (labeled "Kommandozeile Tab: Shell (!)"), showing a terminal window with the command prompt "exit-coding-game\$". Orange arrows point from the text "Datei-Explorer (wie in Windows)" to the file explorer, from "Text-Editor (hier schreibt man den Code)" to the code editor, and from "Kommandozeile Tab: Shell (!)" to the terminal window.

```
exit-hack/03_Texterkennung/examples/hello_world.py
1 def main():
2     print("Hello world!")
3
4 if __name__ == "__main__":
5     main()
6
```

**Demo:** Datei hello\_world.py im Ordner 03\_Texterkennung/examples/ im Editor öffnen



## How To code

**Um ein Python-Programm zu starten muss man den Python Interpreter auf der Kommandozeile starten (der dann wiederum das Python Programm aufruft)**

1. Eintippen von python3 gefolgt von der auszuführenden Datei in der Kommandozeile

```
$ python3 hello_world.py
```

3. Drücken von Enter (**jede Eingabe auf der Kommandozeile mit Enter abschließen!**)

3. Programm wird aufgeführt und die Ausgabe wird erzeugt

```
Hello world!
```

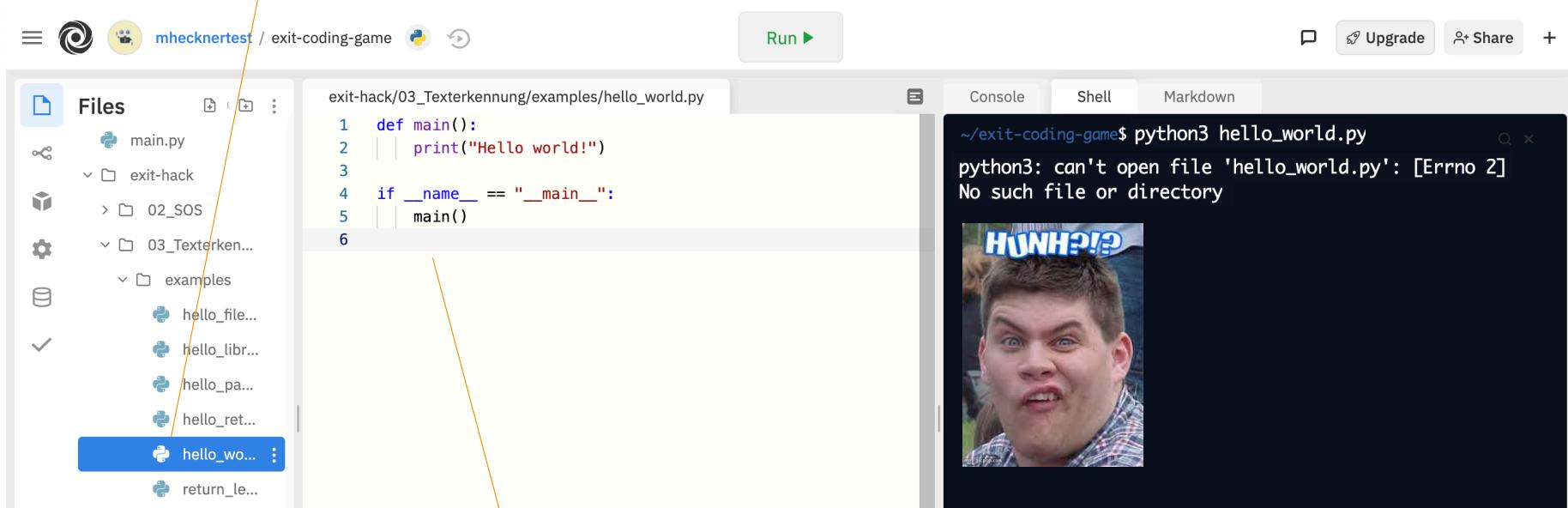


The screenshot shows a Python code editor interface. At the top right is a green 'Run ▶' button. Below it, the file path 'exit-hack/03\_Texterkennung/examples/hello\_world.py' is displayed. The code itself is:

```
1  def main():
2      print("Hello world!")
3
4  if __name__ == "__main__":
5      main()
6
```

# How to code Python – Programme von der Kommandozeile ausführen

Datei hello\_world.py angeklickt



```
exit-hack/03_Texterkennung/examples/hello_world.py
1 def main():
2     print("Hello world!")
3
4 if __name__ == "__main__":
5     main()
6
```

Programmcode in hello\_world.py

Demo: Kommandozeile (wie es nicht geht)



**Auch auf der Kommandozeile muss man sich im  
richtigen Ordner befinden...**

## How to code

# Ordner wechseln auf der Kommandozeile

The screenshot shows a terminal window with the following command history:

```
~/exit-coding-game$ python3 hello_world.py
python3: can't open file 'hello_world.py': [Errno 2]
No such file or directory
~/exit-coding-game$ cd exit-hack
~/exit-coding-game/exit-hack$ cd 03_Texterkennung
~/.../03_Texterkennung$ cd examples
~/.../03_Texterkennung/examples$ python3 hello_world.py
Hello world!
~/.../03_Texterkennung/examples$
```

The terminal window has tabs for "Console", "Shell", and "Markdown". The "Console" tab is selected.

In the center of the window, there is a text overlay that reads:

Jetzt stimmen die Ordner  
Auf Kommandozeile  
Und in der graphischen  
Ansicht überein!

The left side of the window shows a file explorer interface with a sidebar containing icons for Files, Exit-hack, 02\_SOS, 03\_Texterkennung, and a list of files including main.py, hello\_file..., hello\_libr..., hello\_pa..., hello\_ret..., and return\_le... The "examples" folder under 03\_Texterkennung is highlighted with a blue arrow pointing from the text overlay.

Demo: Kommandozeile (wie es richtig geht)



**Während der Bearbeitung des Levels  
bekommt ihr ein Cheatsheet für die  
Kommandozeile...**

## How to code

# Daten an Funktionen übergeben

```
def main():
    message = "Hello World!"
    printMessage(message)

def printMessage(message):
    print(message)

if __name__ == "__main__":
    main()
```

Definiert die Variable `message` und speichert dort den Text "Hello World!"  
In Python gibt es keine Datentypen wie `int`

Variable `message`  
(und deren Inhalt "Hello World!") werden  
an `printMessage` übergeben

`printMessage` erhält den Wert von `message` (aus der  
Funktion `main`) in der neuen Variable `message`.

`print` (in Standard-Python enthalten) gibt  
den Wert von `message` aus.

## How to code

# Funktionen können Werte zurückgeben

```
def main():
    geld = 2
    brotzeit = getLeberkaeseSemmel(geld)
    printBrotzeit(brotzeit)

def getLeberkaeseSemmel(geld):
    print("Kaufe ein für €")
    print(geld)
    return "Leberkäsesemmel"

def printBrotzeit(brotzeit):
    print("Deine Brotzeit: ")
    print(brotzeit)

if __name__ == "__main__":
    main()
```

In brotzeit steht jetzt der zurückgegebene Wert "Leberkäsesemmel"

Variable geld  
(und deren Inhalt 2) werden an getLeberkaeseSemmel übergeben (kennen wir schon)

Variable brotzeit  
Wird an Funktion übergeben (kennen wir auch schon)

Funktion gibt einen Wert zurück (z.B. Ergebnis einer Berechnung, oder Daten, die geholt werden, ...)  
Hier: Ergebnis ist die "Leberkäsesemmel"

Funktion gibt den Inhalt von brotzeit aus, hier:  
"Leberkäsesemmel"



## How to code

# Bibliotheken enthalten Funktionen, die man als Programmierer verwenden kann

```
import time

def main():
    now = time.ctime()
    print("Current Time: ")
    print(now)

if __name__ == "__main__":
    main()
```

Importieren der Bibliothek time.  
Jetzt lassen sich Funktionen aus dieser Bibliothek verwenden.  
D.h. man kann als Entwickler auf der Arbeit anderer Entwickler aufbauen.

Aufrufen der Funktion ctime()  
aus der Bibliothek time

Legt die Variable now an und speichert das Ergebnis der  
Funktion ctime aus der Bibliothek time in der Variable  
now. D.h. die Funktion ctime "gibt Informationen bzw.  
Daten zurück"

