Einleitung

Die ersten Videospiele, die ich im Kindesalter spielte, machten mich süchtig. Ich wollte sie nicht nur spielen, ich wollte selbst so etwas gestalten. Damals fand ich ein Buch, das mir zeigte, wie ich eigene erste Programme und Spiele schreiben konnte. Das machte Spaß und war ganz einfach. Die ersten Spiele, die ich entwickelte, waren ähnlich wie diejenigen in diesem Buch. Sie waren nicht so ausgefeilt wie die Nintendo-Spiele, die meine Eltern für mich kauften, aber sie waren mein eigenes Werk.

Jetzt, als Erwachsener, habe ich immer noch Spaß am Programmieren und werde sogar dafür bezahlt. Doch auch wenn Sie nicht vorhaben, Programmierer zu werden, ist Programmierung eine nützliche und unterhaltsame Fähigkeit. Sie schulen sich dabei, logisch zu denken, planvoll vorzugehen und ihre Ideen kritisch zu betrachten, wenn Sie Fehler in Ihrem Code gefunden haben.

Viele Programmierbücher für Einsteiger fallen in eine von zwei verschiedenen Kategorien. Die erste Art von Büchern konzentriert sich so sehr auf »Spielentwicklungssoftware« oder stark vereinfachende Sprachen, dass man das, was da gelehrt

wird, gar nicht mehr als »Programmieren« bezeichnen kann. Die zweite Kategorie umfasst Bücher, die Programmierung in der Art eines Mathebuchs abhandeln: lauter Lehrsätze und Begriffe, aber nur wenige praktische Anwendungen für die Leser. Das vorliegende Buch verfolgt einen anderen Ansatz und bringt Ihnen das Programmieren am Beispiel von Videospielen bei. Ich stelle Ihnen den Quellcode der Spiele jeweils zu Anfang vor und erkläre dann die Programmierprinzipien anhand dieser Beispiele. Diese Vorgehensweise war für mich selbst von entscheidender Bedeutung, als ich Programmieren lernte. Je mehr ich darüber erfuhr, wie die Programme anderer Leute funktionierten, umso mehr Ideen bekam ich für meine eigenen.

Sie brauchen nicht mehr als einen Computer, den Python-Interpreter (eine kostenlose Software) und dieses Buch. Wenn Sie gelernt haben, wie Sie die Spiele in diesem programmieren, können Sie auch Ihre eigenen entwickeln.

Computer sind unglaubliche Maschinen. Zu lernen, wie man sie programmiert, ist gar nicht so schwer, wie die meisten Leute glauben. Ein Computer*programm* ist eine Folge von Anweisungen, die ein Computer verstehen kann, ähnlich wie ein Buch eine Folge von Sätzen ist, die der Leser verstehen kann. Um einen Computer anzuweisen, etwas zu tun, müssen Sie ein Programm in einer Sprache schreiben, die er versteht. In diesem Buch lernen Sie die Programmiersprache Python kennen. Es gibt noch viele andere Programmiersprachen, die Sie lernen können, z. B. BASIC, Java, JavaScript, PHP oder C++.

Als Kind habe ich BASIC gelernt, aber neuere Programmiersprachen wie Python lassen sich noch einfacher lernen. Python wird auch von professionellen Programmierern sowohl bei der Arbeit als auch für Hobbyprojekte verwendet. Installation und Verwendung sind außerdem völlig kostenlos. Sie brauchen lediglich eine Internetverbindung, um die Sprache herunterzuladen.

Da Videospiele nichts anderes sind als Computerprogramme, bestehen sie ebenfalls aus Anweisungen. Im Vergleich zu den Spielen für die Xbox, die PlayStation oder den Nintendo wirken diejenigen, die Sie in diesem Buch erstellen werden, ziemlich simpel. Sie weisen keine anspruchsvolle Grafik auf, sondern sind absichtlich einfach gehalten, sodass Sie sich drauf konzentrieren können, Programmieren zu lernen. Außerdem müssen Spiele nicht kompliziert sein, um Spaß zu machen.

Zielgruppe

Programmieren an sich ist nicht schwer. Dagegen kann es durchaus schwer sein, Anleitungen zu finden, die einem sagen, wie man interessante Dinge programmiert. Andere Computerbücher behandeln viele Themen, die Neulinge gar nicht brau-

chen. Dieses Buch zeigt Ihnen, wie Sie Ihre eigenen Spiele programmieren. Damit erwerben Sie eine nützliche Fähigkeit und haben außerdem unterhaltsame Spiele, die Sie stolz vorstellen können. Dieses Buch richtet sich an folgende Leser:

- Anfänger ohne jegliche Vorkenntnisse, die sich selbst das Programmieren beibringen möchten
- Kinder und Jugendliche, die durch das Schreiben von Spielen programmieren lernen möchten
- Erwachsene und Lehrer, die anderen Programmieren beibringen möchten
- Jegliche Personen, ob jung oder alt, die anhand einer professionellen Programmiersprache programmieren lernen wollen

Der Aufbau dieses Buches

In den meisten Kapiteln dieses Buches wird jeweils ein neues Spielprojekt vorgestellt und erläutert. Einige wenige Kapitel sind zusätzlichen interessanten Themen gewidmet, z. B. dem Debugging. Neue Programmierprinzipien werden jeweils dann vorgestellt, wenn sie in den Spielen vorkommen. Die Kapitel sollten in der vorliegenden Reihenfolge gelesen werden. Die folgende kurze Aufstellung zeigt, was Sie in den einzelnen Kapiteln erwartet:

Kapitel 1: Die interaktive Shell erklärt, wie Sei die interaktive Shell von Python verwenden können, um Code zeilenweise auszuprobieren.

Kapitel 2: Programme schreiben erklärt, wie Sie im Dateieditor von Python vollständige Programme schreiben.

In **Kapitel 3: Zahlen raten** programmieren Sie das erste Spiel in diesem Buch. Dabei muss der Spieler eine Geheimzahl raten und erhält jeweils den Hinweis, ob seine Vermutung zu hoch oder zu niedrig war.

In **Kapitel 4: Ein Kalauerprogramm** schreiben Sie ein einfaches Programm, das dem Benutzer mehrere Flachwitze erzählt.

In **Kapitel 5: Im Reich der Drachen** programmieren Sie ein Ratespiel, in dem der Spieler zwischen zwei Höhlen wählen muss: In einer der beiden wohnt ein freundlicher Drache, in der anderen ein hungriger.

Kapitel 6: Der Debugger erklärt, wie Sie Probleme in Ihrem Code mithilfe des Debuggers lösen.

Kapitel 7: Galgenmännchen: Entwurf mit einem Flussdiagramm beschreibt, wie Sie Flussdiagramme zur Planung anspruchsvollerer Programme einsetzen, hier für das in den folgenden Kapiteln beschriebene Galgenmännchen-Spiel.

In **Kapitel 8: Galgenmännchen: Der Code** schreiben Sie das Galgenmännchen-Spiel anhand des Flussdiagramms aus Kapitel 7.

In **Kapitel 9: Galgenmännchen: Erweiterungen** erweitern Sie das Galgenmännchen-Spiel, indem Sie den Python-Datentyp der Dictionarys verwenden.

In **Kapitel 10: Tic-Tac-Toe** lernen Sie, wie Sie ein Tic-Tac-Toe-Spiel schreiben, bei dem eine künstliche Intelligenz gegen den menschlichen Benutzer antritt.

In **Kapitel 11: Bagels** schreiben Sie das Denksportspiel Bagels, in dem der Spieler anhand von Hinweisen eine Geheimzahl erraten muss.

Kapitel 12: Kartesische Koordinaten erklärt das kartesische Koordinatensystem, das Sie in nachfolgenden Spielen verwenden werden.

In **Kapitel 13: Sonar-Schatzsuche** schreiben Sie ein Spiel, in dem Sie das Meer nach verlorenen Schatztruhen absuchen.

In **Kapitel 14: Die Caesar-Chiffre** schreiben Sie ein einfaches Verschlüsselungsprogramm, mit dem Sie Geheimnachrichten verschlüsseln und entschlüsseln können.

In **Kapitel 15: Reversegam** programmieren Sie ein anspruchsvolles Reversi-ähnliches Spiel mit einer fast unschlagbaren künstlichen Intelligenz als Gegner.

Kapitel 16: Reversegam: AI-Simulation erweitert das Reversegam-Speil aus Kapitel 15, sodass nun verschiedene KIs gegeneinander antreten.

Kapitel 17: Grafik einsetzen führt das Python-Modul `pygame` ein und zeigt, wie Sie damit 2-D-Grafiken zeichnen können.

Kapitel 18: Grafiken animieren zeigt, wie Sie mit `pygame` Grafiken animieren.

In **Kapitel 19: Kollisionserkennung** lernen Sie, wie Sie in 2-D-Spielen Zusammenstöße zwischen Objekten erkennen können.

In **Kapitel 20: Sounds und Bilder verwenden** erweitern Sie Ihre einfachen `pygame`-Spiele um Klänge und Bilder.

Kapitel 21: Dodger mit Ton und Grafik wendet den Stoff aus den Kapiteln 17 bis 20 für ein animiertes Spiel namens Dodger an.

Die Codebeispiele

Die meisten Kapitel dieses Buches beginnen mit einem Beispieldurchlauf des Programms, das in dem Kapitel vorgestellt wird. Dieser Durchlauf zeigt, wie das Programm aussieht, wenn Sie es ausführen. Die Eingaben des Benutzers sind durch Fettdruck gekennzeichnet.

Ich rate Ihnen, den Code der einzelnen Programme jeweils selbst in den IDLE-Dateieditor einzugeben, anstatt ihn einfach nur herunterzuladen und zu kopieren. Wenn Sie sich die Zeit nehmen, den Code abzutippen, werden Sie mehr davon im Kopf behalten.

Zeilennummern und Einrückungen

Beim Abtippen des Quellcodes aus dem Buch dürfen Sie die Nummern zu Beginn der einzelnen Zeilen *nicht* mit eingeben! Nehmen wir an, Sie sehen die folgende Codezeile:

```
9. number = random.randint(1, 20)
```

Hier dürfen Sie weder `9.` noch das darauffolgende Leerzeichen eingeben, sondern nur Folgendes:

```
number = random.randint(1, 20)
```

Die Zeilennummern stehen hier nur, um in den Erklärungen auf die einzelnen Zeilen verweisen zu können. Sie gehören nicht zum Quellcode des Programms.

Abgesehen von diesen Zeilennummern jedoch müssen Sie den Code genau so eingeben, wie er in dem Buch erscheint. Einige Zeilen sind mit vier, acht oder mehr Leerzeichen eingerückt. Diese Leerzeichen zu Anfang einer Zeile wirken sich darauf aus, wie Python die Anweisungen deutet, weshalb es sehr wichtig ist, sie einzuschließen.

Betrachten Sie dazu das folgende Beispiel. Die Leerzeichen sind durch dicke Punkte (•) dargestellt, sodass Sie sie sehen können:

```
while guesses < 10:
····if number == 42:
········print('Hello')
```

Die erste Zeile ist nicht eingerückt, die zweite um vier und die dritte um acht Leerzeichen. In den tatsächlichen Beispielen in dem Buch sind die Leerzeichen zwar nicht durch solche Punkte gekennzeichnet, aber da alle Zeichen in IDLE die gleiche Breite aufweisen, können Sie die Anzahl der Leerzeichen einfach dadurch

bestimmen, dass Sie die Anzahl der Zeichen in der darüber- oder darunterliegenden Zeile zählen.

Lange Codezeilen

Manche Codeanweisungen sind so lang, dass sie nicht mehr in eine Buchzeile passen und daher in die nächste Codezeile umbrechen. Allerdings passt die Zeile auf den Computerbildschirm. Geben Sie sie dort in einer einzigen Zeile ein, ohne zwischendurch die Eingabetaste zu drücken. Ob eine Anweisung einfach nur fortgesetzt wird oder ob tatsächlich eine neue Anweisung beginnt, können Sie anhand der Zeilennummern auf der linken Seite erkennen. Das folgende Beispiel zeigt nur zwei Anweisungen:

```
1. print('This is the first instruction!xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
     xxxxxxxxxxxx')
2. print('This is the second instruction, not the third instruction.')
```

Die erste Anweisung bricht in die zweite Buchzeile um. Da die zweite Buchzeile keine Zeilennummer trägt, können Sie erkennen, dass es sich immer noch um die erste Codezeile handelt.

Python herunterladen und installieren

Um die Programme eingeben und ausprobieren zu können, Sie müssen eine Software installieren, die als Python-*Interpreter* bezeichnet wird und die Anweisungen versteht, die Sie in der Sprache Python geschrieben habe. Den Python-Interpreter werde ich im Folgenden der Einfachheit halber auch einfach als Python bezeichnen.

In diesem Abschnitt zeige ich Ihnen, wie Sie Python 3 – genauer gesagt, Python 3.4 – für Windows, OS X und Ubuntu herunterladen. Es gibt schon neuere Versionen als 3.4, aber das Modul `pygame`, das wir in den Kapiteln 17 bis 21 verwenden, unterstützt zurzeit nur die Versionen bis 3.4.

Beachten Sie auch, dass es erhebliche Unterschiede zwischen Python 2 und Python 3 gibt. Die Programme in diesem Buch verwenden Python 3. Wenn Sie versuchen, Sie mit Python 2 auszuführen, erhalten Sie Fehlermeldungen. Das ist so wichtig, dass ich bei jedem Programm den kleinen Pinguin hinzugefügt habe, um Sie darauf aufmerksam zu machen.

Wenn Sie mit Windows arbeiten, laden Sie den MSI-Installer für Windows x86-64 von *https://www.python.org/downloads/release/python-344/* herunter und doppelklicken darauf. Möglicherweise müssen Sie das Administratorkennwort für Ihren Computer eingeben.

Folgen Sie den Anweisungen, die der Installer anzeigt, um Python zu installieren. Die Vorgehensweise ist wie folgt:

1. Wählen Sie *Install for All Users* und klicken Sie auf *Weiter*.
2. Bestätigen Sie die Installation im Ordner *C:\Python34*, indem Sie auf *Weiter* klicken.
3. Klicken Sie auf *Weiter*, um den Abschnitt *Customize Python* zu überspringen.

Wenn Sie mit OS X arbeiten, laden Sie den MSI-Installer für Mac OS X 64 Bit/32 Bit von *https://www.python.org/downloads/release/python-344/* herunter und doppelklicken darauf. Möglicherweise müssen Sie das Administratorkennwort für Ihren Computer eingeben.

Folgen Sie den Anweisungen, die der Installer anzeigt, um Python zu installieren. Die Vorgehensweise ist wie folgt:

1. Wenn Sie eine Warnung der Art »Python.mpkg kann nicht geöffnet werden, da es von einem nicht verifizierten Entwickler stammt« erhalten, klicken Sie bei gedrückter ctrl-Taste auf die Datei *Python.mpkg* und wählen *Öffnen* aus dem dann angezeigten Menü. Möglicherweise müssen Sie das Administratorpasswort für Ihren Computer eingeben.
2. Klicken Sie im Willkommensabschnitt auf *Continue* und dann auf *Agree*, um die Lizenzverarbeitung zu akzeptieren.
3. Wählen Sie *Macintosh HD* (bzw. den Namen Ihrer Festplatte) aus und klicken Sie auf *Install*.

Wenn Sie Ubuntu verwenden, könne Sie Python wie folgt aus dem Ubuntu Software Center heraus installieren:

1. Öffnen Sie das Ubuntu Software Center.
2. Geben Sie in das Suchfeld oben rechts in dem Fenster **Python** ein.
3. Wählen Sie *IDLE (Python 3.4 GUI 64 bit)* aus.
4. Klicken Sie auf *Install*. Möglicherweise müssen Sie das Administratorpasswort Computer eingeben, um die Installation abzuschließen.

Sollten die angegebenen Vorgehensweisen nicht funktionieren, finden Sie alternative Installationsanweisungen für Python 3.4 auf *https://www.nostarch.com/inventwithpython/*.

IDLE starten

IDLE steht für Interactive Development Enrivonment, also »interaktive Entwicklungsumgebung«. Es handelt sich dabei um eine Art Textverarbeitungssystem zum Schreiben von Python-Programmen. Je nach Betriebssystem müssen Sie zum Starten dieser Software jeweils unterschiedlich vorgehen:

- Unter Windows klicken Sie auf das Startmenü unten links, geben **IDLE** ein und wählen *IDLE (Python GUI)*.
- Unter OS X öffnen Sie den Finder und klicken auf *Programme*. Doppelklicken Sie anschließend auf *Python 3.x* und dann auf das IDLE-Symbol.
- Unter Ubuntu und anderen Linux-Distributionen öffnen Sie ein Terminal-Fenster und geben **idle3** ein. Es kann auch sein, dass Sie auf *Applications* am oberen Bildschirmrand und dann auf *Programming* und *IDLE3* klicken können.

Das Fenster, das Sie beim Start von IDLE sehen, ist die *interaktive Shell* (siehe Abbildung). Wenn Sie hinter der Eingabeaufforderung >>> Python-Anweisungen in die Shell eingeben, führt Python sie aus. Anschließend erscheint wieder die Eingabeaufforderung >>>, um auf Ihre nächste Anweisung zu warten.

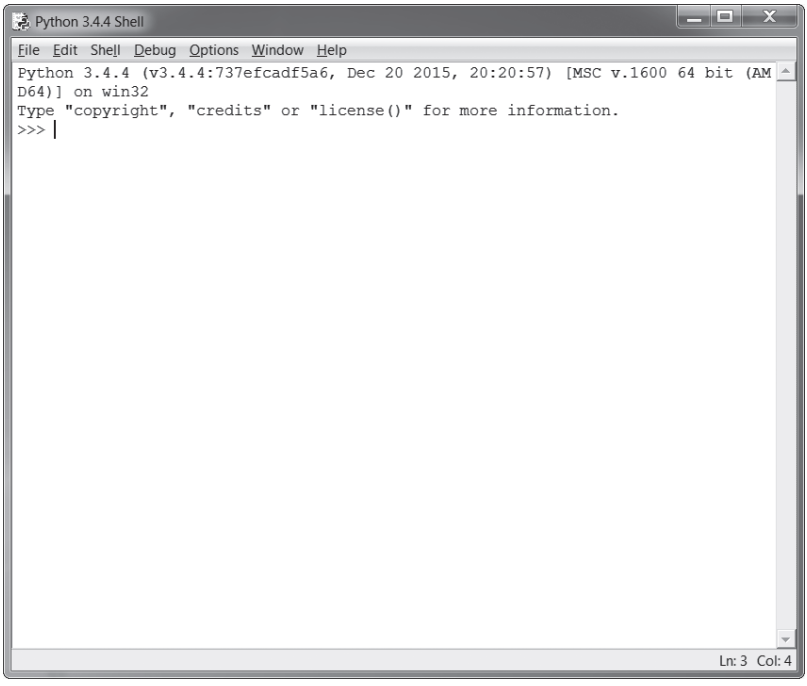

Abb. E–1 *Die interaktive Shell von IDLE*