

1 Einleitung

*Ich denke nicht in Kategorien wie Bewegungsspiel,
Brettspiel oder Videospiel. Für mich gibt's nur
die Kategorien »Macht Spaß« und »Macht keinen Spaß«.*

Keita Takahashi aus »/GameStar/dev« 01/2006

*..»and what is the use of a book,« thought Alice,
»without pictures or conversations?«*

Alice's Adventures in Wonderland

Das Mobiltelefon ist in den letzten zehn Jahren von einem Requisite in Agentenfilmen zu einem Alltagsgegenstand geworden. Viele Menschen benutzen es heutzutage nicht nur zum Telefonieren, sondern auch zum Fotografieren, Musikhören oder natürlich zum Spielen.

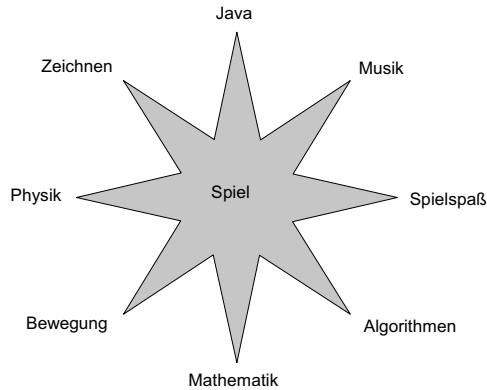
Da immer mehr Personen gerade in Situationen, in denen sie auf etwas warten müssen, zum Mobiltelefon als Computerspielersatz greifen, hat sich das Geschäft mit Handyspielen zu einem milliarden-schweren Markt weltweit entwickelt.

Wenn Sie nicht nur Spaß haben, damit zu spielen, und dadurch als Konsument am Markt teilnehmen, sondern sich auch schon einmal gedacht haben: »Wie funktioniert ein Handyspiel eigentlich? Kann ich selbst eins programmieren?«, dann sind Sie bei diesem Buch richtig.

1.1 Was Sie in diesem Buch erwartet

Das Interessante an der Spieleprogrammierung ist, dass Wissen aus ganz unterschiedlichen Bereichen, wie Grafik, Musik oder Softwareentwicklung, für ein gutes Endprodukt kombiniert werden muss.

Abb. 1-1
Basiswissen für ein
Handyspiel



Dieses Buch setzt neben grundsätzlichen Java-Programmierkenntnissen keinerlei Fachwissen beim Leser voraus. Es vermittelt vielmehr einen Einstieg in diese verschiedenen Gebiete.

Kapitelübersicht

Kapitel 1 zeigt, wie man ein eigenes Programm auf einem Handy zum Laufen bringt und wie es zum heutigen Stand der Java-Technik im mobilen Bereich gekommen ist.

Kapitel 2 liefert das erste komplette Handyspiel: ein einfaches Ping-Pong-Spiel. Der grundlegende Aufbau eines Spiels und das dafür benötigte Grafikkwissen sind die Hauptthemen.

Kapitel 3 baut die Grafikprogrammierung aus, damit das Spiel Freddy-TheFish, ein einfaches Jump'n'Run-Spiel, visuell an kommerzielle Vorbilder herankommt. Dazu wird die Java Game API genauer unter die Lupe genommen.

Kapitel 4 beschäftigt sich mit der Programmoberfläche, die um ein Spiel herum benötigt wird. Hier geht es um Highscore-Listen oder wie der Spieler bestimmte Optionen im Spiel verändern kann.

Kapitel 5 liefert den Sound zum Spiel. Es wird ausgelotet, wie das Spielerlebnis durch Musik und Geräusche verbessert werden kann.

Kapitel 6 stellt die Verbindung zu anderen Mobiltelefonen her. Die Integration von mehreren Mitspielern in einem Spiel ist sowohl übertragungstechnisch als auch von der Spiellogik her keine einfache Aufgabe.

Kapitel 7 ermöglicht den Einstieg in die Welt der 3D-Grafik, wie sie jeder von seinem PC kennt. Ein 3D-Murmespiel mit Licht und Schattierungen ist das Ergebnis.

Kapitel 8 rundet das 3D-Spiel mit der entsprechenden Physik und einer automatischen Kameraführung ab.

Für alle Beispielprogramme in diesem Buch reichen grundlegende Java-Kenntnisse aus. Alle tiefer gehenden Aspekte werden in den einzelnen Kapiteln ausführlich erläutert. Falls Sie mehr technische Informationen suchen, werden Sie im Buch »Java Micro Edition« von Klaus-Dieter Schmatz fündig [Sch07].

1.2 Übersicht: Java auf dem Handy

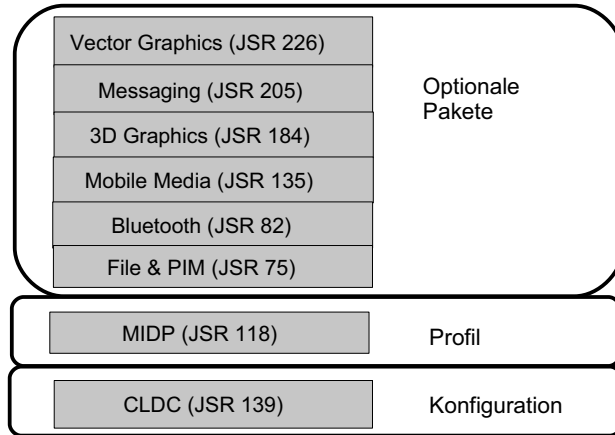
Zu Beginn war Java mit dem Schlagwort »WORA« (Write Once Run Anywhere) – »Einmal schreiben – läuft überall« angetreten. Irgendwann hat die Firma Sun, einer der Erfinder von Java, festgestellt, dass die Geräte, auf denen die Programme laufen sollen, doch zu verschiedenen sind, und sich für drei Editionen von Java entschieden:

- *Java Enterprise Edition (Java EE)* ist für unternehmensweite Anwendungen mit sehr vielen Anwendern geeignet. Sie enthält viele Erweiterungen zur Serverprogrammierung.
- *Java Standard Edition (Java SE)* ist für den gängigen PC mit Betriebssystem OS X, Linux, Solaris oder Microsoft Windows gedacht.
- *Java Micro Edition (Java ME)* wurde speziell für kleine Geräte mit wenig Speicher und Rechenleistung entwickelt. Darunter fallen natürlich Mobiltelefone, aber auch PDAs, Empfangsgeräte für Fernseher (engl. *set-top box*) oder Steuerungsmodule in Autos.

Für dieses Sammelsurium von Kleingeräten und deren verschiedene Anwendungsbereiche sind in Java ME eine Unmenge von Programmibliotheken (API, engl. *Application Programming Interface*) vorhanden.

Um etwas Ordnung in dieses Sammelsurium von APIs zu bekommen, gibt es für jede Gerätekategorie eine sogenannte Konfiguration, ein Profil und darüber hinaus optionale Pakete.

Abb. 1-2
Konfiguration, Profil und
optionale Pakete



Die **Konfiguration** beinhaltet eine virtuelle Maschine und einige grundlegende Bibliotheken. Bei aktuellen Handys gibt es die Konfigurationen »Connected Limited Device Configuration (CLDC)« oder »Connected Device Configuration (CDC)«. CLDC ist im Groben eine Teilmenge von CDC und reicht für alle Beispiele in diesem Buch vollkommen aus.

Das **Profil** vervollständigt die Konfiguration mit zusätzlichen APIs zu einer kompletten Plattform, auf der Programme laufen können. »Mobile Information Device Profile (MIDP)« ist das Profil für Handys. Aktuell ist die Version 2.x.

Die **optionalen Pakete** erweitern die grundlegende Funktionalität um ganz spezielle APIs, wie 3D-Grafik, Kommunikation über Bluetooth oder andere, die nicht bei jedem Handy vorhanden sind. Die in Abb. 1-1 aufgelisteten sind nur die wichtigsten Pakete, die auch auf den meisten Geräten zur Verfügung stehen.

Alle Java ME-Spezifikationen oder APIs haben durch ihren Entstehungsprozess, den *Java Specification Request* (JSR), eine eindeutige Nummer. Um die APIs einfacher unterscheiden zu können, wird häufig diese JSR-Nummer mit angegeben.

1.3 Wie alles begann

Die Geschichte der Handyspiele ist relativ kurz. Alles begann mit Herrn Taneli Armanto, der 1997 für das Nokia 6110 ein Spiel mit dem Namen »Snake« entwickelte. Ein Handy zum Spielen ist heutzutage eine Selbstverständlichkeit, doch damals war es ein weitsichtiger Gedanke, in ein teures Hightech-Gerät für Geschäftskunden ein Spiel zu integrieren. Er legte damit den Grundstein für einen Industriezweig,