Hartmut Gieselmann

Wir machen nur Spaß

Ihr Einstieg in die Spiele-Entwicklung

Der Boom der Indie-Spiele hat zu einer Neubelebung der Entwicklerszene geführt. Nie war es einfacher, sein eigenes Spiel zu entwickeln und zu vertreiben – auch ohne großen Publisher: Die Programmier-Werkzeuge sind oftmals kostenlos erhältlich, vertreiben lassen sich die fertigen Spiele über eine Webseite, App-Stores oder Download-Portale. Da braucht es nur noch eine zündende Idee.

Als angehender Spiele-Entwickler sollten Sie sich nicht von großen Namen in den Verkaufshitlisten abschrecken lassen. Neben Skyrim & Co., an denen Hunderte von Entwicklern über Jahre gearbeitet haben, gibt es auch viele kleine Spiele, die Hobby-Programmierer an einem Wochenende schreiben und damit auf Indie-Festivals ihre Fans finden. Wir stellen solche kleinen kreativen Perlen regelmäßig in den Indie- und Freeware-Tipps und auf heise.de in der Video-Rubrik „c't zockt" vor. Keine Angst also, wenn Sie alleine sind, nur nach Feierabend oder Schulschluss an ihrem Spiel arbeiten und parallel noch eine Programmiersprache lernen oder sich in eine Entwicklungsumgebung einarbeiten. Gehen Sie einfach ein kleines überschaubares Projekt an, das sich auf eine besondere Spielmechanik konzentriert. Bevor wir Ihnen aber verschiedene Entwicklungswerkzeuge vorstellen, zeigen wir einige simple Beispiele, wie Sie auf neue Ideen kommen, und stellen Ihnen im Anschluss ein kleines konkretes Projekt vor, damit Sie gleich loslegen können.

Um sich mit der Materie vertraut zu machen, können Sie sich an der Evolution der Videospiele orientieren: Nach ersten Fingerübungen mit einem TicTacToe fängt man mit einem simplen Beispiel wie Pong an, bei dem man lernt, Eingaben des Spielers in Bewegungen auf dem Bildschirm umzusetzen, Kollisionen abzufragen und Punkte für den High-Score zu zählen. Später programmieren Sie vom Computer gesteuerte Gegner in einfachen 2D-Spielen wie unseren Pac-Man-Klon PieGuy ab Seite 110. Bei diesen lernen Sie den Umgang mit Sprites und Sounds. In der nächsten Stufe könnte man sich etwa an scrollenden Bildschirmhintergründen wie in Super Mario Bros. probieren. So lernen Sie die Grundfertigkeiten der Spieleentwicklung beim Nachbau von Retro-Klassikern und bekommen das Rüstzeug für Ihre erste eigene Idee.

Weniger ist mehr

Die teuren Produktionen protzen mit detaillierten 3D-Grafiken, aufwendigen Zwischensequenzen, anpassbaren Spielcharakteren, herunterladbaren Zusatz-

paketen, Twitter- und Facebook-Meldungen – doch so ein Pomp bläht das eigentliche Spiel nur auf. Kein Wunder, dass sich viele Spieler die gute alte Zeit der 80er und 90er Jahre zurückwünschen, wo alles einfacher war. Und genau diesem Wunsch nach Simplifizierung können Hobby-Programmierer und Indie-Entwickler nachkommen und ihren Mangel an Zeit und Geld in eine Stärke gegenüber den großen Studios umwandeln.

Analogien dazu findet man auch in der Spielzeugwelt: „Gutes" Spielzeug besteht bekanntlich aus Holz – ein einfacher Klotz lässt sich mehrfach verwenden, er kann ein Auto, ein Kühlschrank oder ein Haus sein, je nachdem, wie das Kind ihn gerade einsetzt. Fehlende Details werden von der Phantasie ergänzt. Plastikspielzeug ist hingegen bis ins letzte Detail ausformuliert: es gibt rote Feuerwehrautos, gelbe Rennboote, Polizeistationen, Weltraumgleiter – doch jedes Element kann nur in dem vom Hersteller vorgesehenen Kontext eingesetzt werden. Will man etwas anderes spielen, muss man etwas Neues kaufen.

Als Privatentwickler müssen Sie ebenfalls mit ihren Ressourcen haushalten. Also orientieren Sie sich am einfachen Holzspielzeug und nicht am komplexen Plastik. So sollte man sich schon während der Konzeptionsphase eines neuen Spiels überlegen, welche Elemente (Texturen, Objekte, Sounds) man tatsächlich benötigt und welche man im Idealfall mehrfach verwenden kann. Vermeiden Sie Doppelungen. Vermeiden Sie Doppelungen. Wenn Sie Einheiten für ein Strategiespiel kreieren, reduzieren sie die Vielfalt der Einheiten auf das Nötigste. Sie brauchen nur einen Nahkämpfer, einen Schützen und einen Heiler, nicht noch einen heilenden Schützen oder einen pfeileschießenden Ritter. Ebenso begnügt man sich in Shootern mit nur einer Pistole und einem Gewehr – es braucht keine 20 verschiedenen Ballermänner. Verwenden Sie einfache Grafiken, die die Elemente unterscheidbar machen und überlassen sie die Details der Phantasie des Spielers – ganz nach dem alten Designer-Credo: Perfektion ist, wenn man nichts mehr weglassen kann.

Ein Paradebeispiel für diesen Ansatz ist „Thomas was alone", ein Jump & Run, dessen Protagonisten nur aus farbigen Rechtecken bestehen. Mit Leben und Persönlichkeit füllt sie ein Erzähler, der während des Spiels den geometrischen Figuren Namen gibt und ihre Geschichte erzählt. Auf aufwendig gerenderte Charaktere kann das Spiel verzichten.

Diesen Grundsatz verletzen jedoch viele Entwickler, weil sie glauben, ihr Spiel sei erst perfekt, wenn sie nichts mehr hinzufügen können. Ubisofts Assassin's Creed ist hier ein unrühmliches Beispiel: Aus dem ursprünglichen Kletter- und Schleichspiel ist inzwischen eine Seefahrer-Simulation mit Echtzeitstrategie-Elementen geworden – aus Designer-Sicht eine Katastrophe. Als Indie-Entwickler sollten Sie genau den anderen Weg gehen und alle Elemente aussieben, die keinen Spaß machen und den Kern so lange polieren, bis er richtig glänzt.

Isolieren, Abstrahieren, Kombinieren

Doch während Spieleentwickler speziell in Deutschland ihre Kunst häufig mehr als technisches Handwerk verstehen und erst einmal einen Business-Plan ausarbeiten, bevor sie ein eigenes Spiel auf den Markt bringen, gehen Indie-Entwickler im Ausland wesentlich freier an die Sache heran. Sie versuchen nicht, ein bereits erfolgreiches Spiel in einem etablierten Genre nachzuahmen, sondern neue Wege zu gehen. Natürlich kann man sich an etablierten Genre-Elementen orientieren, weil diese bereits bewiesen haben, dass sie Spaß machen. Aber man sollte sie in einem neuen Spiel nicht bloß imitieren.

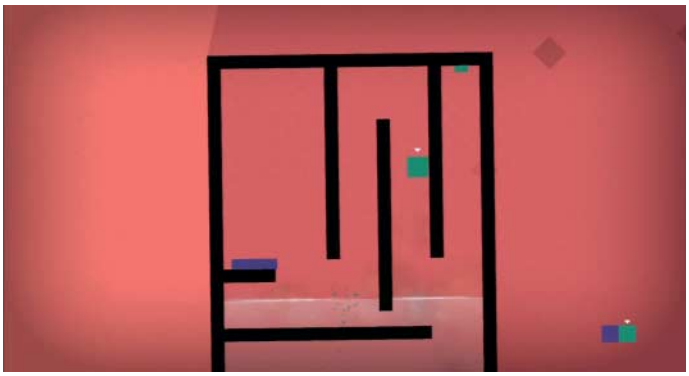
Genres sind gut, wenn ein Mediamarkt-Mitarbeiter eine Spielebox ins passende Regal räumen soll. Für den kreativen Prozess bei der Entwicklung eines Spiels sind Genre-Grenzen jedoch Denkbarrieren, die es einzureißen gilt. James Lantz, Spiele-Designer des Roguelike Mercury, stellte deshalb auf der Game Developers Conference vier Methoden vor, wie man aus einer Mixtur von bestehenden Genre-Vorgaben etwas Neues schaffen kann.

So kann man eine bestimmte Spielmechanik isolieren und zu einem komplett neuen Spiel ausweiten. Auf diese Art haben sich etwa die Tower-Defense-Spiele aus dem Genre der Echtzeitstrategie entwickelt, indem sie sich auf den Bereich Aufbau und Verteidigung der eigenen Basis konzentrierten. Ebenso hat „Canabalt" aus dem Genre der Jump & Runs sich einzig auf den Aspekt des richtigen Abspringens konzentriert, während die Figur automatisch immer weiter läuft.

Eine zweite Methode ist, die Spielmechaniken eines Genres zu reduzieren und auf einer höheren Abstraktionsebene neu zusammenzusetzen. So begnügt sich „Galcon" mit einem einzigen Einheitentyp, mit dem man schnell neue Planeten mit einem Fingertipp erobern oder gegenüber Angreifern verteidigen muss. Produktion, Angriff und Verteidigung wurden hier auf ihren Kern reduziert und zu einem neuen, fast Arcade-artigen Spiel zusammengesetzt. Bei den Jump & Runs lässt „VVVVVV" nicht die Spielfigur springen, sondern den Spieler die Richtung der Schwerkraft ändern, um Hindernissen auszuweichen.

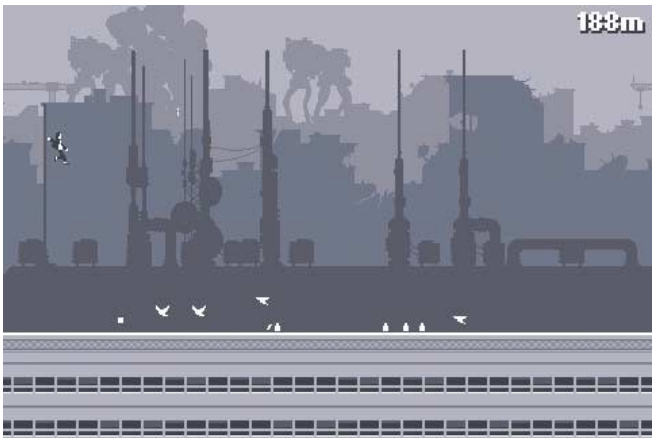
Oder aber man nimmt sich ein wesentliches Genre-definierendes Element vor und ändert es ab, dreht es komplett um. „Braid" spielte mit der Grundregel, dass die Zeit in Jump & Runs immer vorwärts laufen muss, und änderte sie ab, um eine neue Form von Puzzles zu schaffen.

Zu guter Letzt lassen sich natürlich Elemente aus verschiedenen Genres auseinandernehmen und neu kombinieren. Selbst ein Matchmaking-Spiel wie „Bejewe-

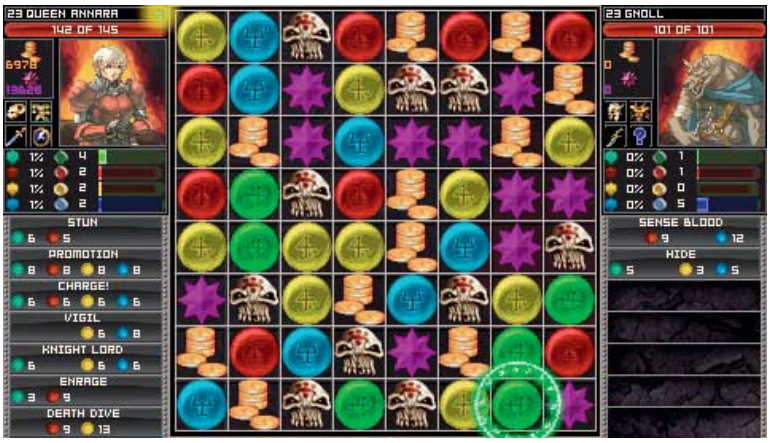


In „Thomas was alone" haucht eine Erzählerstimme einfachen geometrischen Figuren Leben ein. Durch seinen ressourcenschonenden Umgang mit Spielelementen stellt es seinen spielerischen Kern mustergültig heraus.

Assassin's Creed blähte seinen ursprünglichen Kern bis zur Unkenntlichkeit auf. Statt Schleichen und Klettern bestimmen Seeschlachten, Strategie- und Wirtschafts-Elemente die neuesten Serienfolgen.



Canabalt reduzierte Jump & Runs alleinig auf die Reaktionstests beim Abspringen und baute dieses Prinzip zu einem kompletten Spiel aus.



Puzzle Quest kombinierte zwei unterschiedliche Genres und ließ seine Rollenspiel-Kämpfe als ein Matchmaking im Stile von Bejeweled ablaufen.

led“ verträgt sich in „Puzzle Quest“ mit Rollenspiel-Elementen. Allerdings funktioniert das nicht immer: Als Nintendo in „Odama“ einen Flipper mit Echtzeitstrategie kreuzte, kam eine kaum steuerbare, spaßfreie Chimäre heraus.

Ebenso hat Namcos Pac-Man inzwischen viele Entwickler inspiriert, das Labyrinth-Thema zu remixen. Es gibt Versionen, die das Spiel mit einem Ego-Shooter kreuzen oder in denen man nicht den Pac-Man bewegt, sondern das Labyrinth dreht. Es gibt eine Brettspiel-Version, in denen Pac-Man und die Geister abwechselnd ziehen oder Wettbewerbs-Varianten, in der man in einer vorgegebenen Zeit möglichst viele Pillen fressen muss. Man könnte auch den Spieler die vier Geister steuern lassen, um Pac-Man zu fangen. Eine Auswahl der Variationsmöglichkeiten, die wir bereits in c’t-zockt vorgestellt haben, finden Sie in einem kurzen Video-Clip über den Link am Ende des Artikels. Ihnen fallen aber bestimmt noch viele weitere Möglichkeiten ein.

Pappe statt Pixel

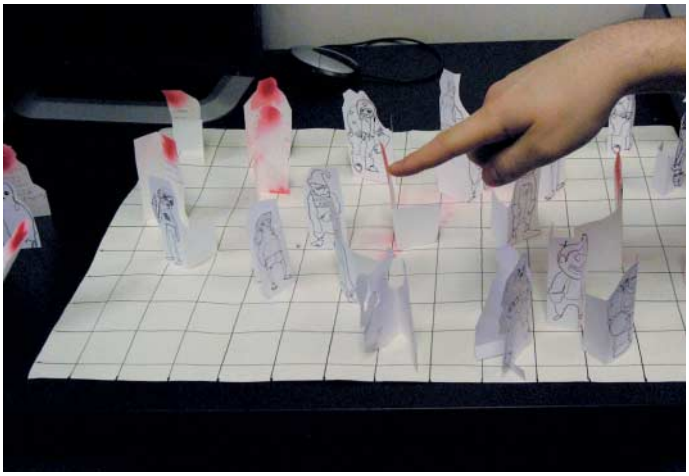
Wenn Ihre erste Spielidee steht, sollten Sie sie an Prototypen ausprobieren. Diese bestehen aus möglichst einfachen Elementen, mit denen man den Kern der Spielmechanik testen kann. Das muss nicht unbedingt in Software passieren, sondern kann auch mit einem Plan aus Pappe, Glasperlen und Holzfiguren aus einer Spielesammlung aufgebaut werden. Wenn Sie die einzelnen Spielelemente erst einmal tatsächlich aus Pappe, Holz, Schere und Kleber aufbauen, merken Sie ganz schnell, was Sie tatsächlich benötigen, und sehen, ob Ihre Spielidee funktioniert oder abgeändert werden muss. „Fail Fast“ heißt hier die Maxime: Was nicht funktioniert, wird verworfen und die nächste Idee gleich an einem neuen Prototyp ausprobiert. Iterieren Sie so lange, bis es Spaß macht.

Das gilt nicht nur für Strategiespiele, selbst Level für Ego-Shooter kann man zunächst aus Pappe aufbauen, um zu sehen, ob die Räume genügend Deckungsmöglichkeiten bieten und wo man die Gegner am besten platziert. Wenn Sie im Team arbeiten, können an einem solch gebastelten Prototypen alle Beteiligten sehen und ausprobieren, wie das eigentliche Spiel später funktionieren soll.

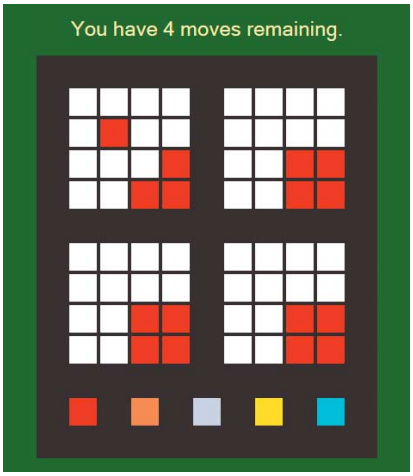
Der Umgang mit realen Spielmaterialien hat selbst für etablierte Spieldesigner noch immer einen besonderen Reiz. Industrie-Veteranen wie Brenda Romero setzen manche Ideen sogar nur als Brettspiel um. Diese Prototyp-Methode propagiert etwa Tracy Fullerton in ihrem empfehlenswerten (englischen) Buch „Game Design Workshop“. Bei Fullerton lernten beispielsweise Nova Chen und Kellee Santiago, die als thatgamecompany Spiele wie „Flower“ und „Journey“ entwickelten. Sie probierten Dutzende von Prototypen, bis sie schließlich auf das finale Spielprinzip dieser inzwischen als Klassiker geltenden Indie-Spiele kamen.

Haben Sie ihren Papp-Prototyp soweit fertig und das Regelwerk abgestimmt, können Sie sich an die Umsetzung eines digitalen Prototyps machen. Dazu muss man nicht gleich selbst coden, sondern kann sich einer bereits fertigen Spiele-Engine bedienen. Jedoch sollten Sie darauf achten, dass Ihre Idee nicht unter den Beschränkungen der Prototyp-Engine leidet. Manchmal können aber auch gerade solche Grenzen inspirieren.

Eine besonders einfache Engine, die Brian Moriarty, Professor am Worcester Polytechnic Institute und ehemaliger Entwickler



Wenn der Prototyp einer Spielidee selbst mit einfachen Papierfiguren auf einem aufgemalten Gitter Spaß macht, dann ist er reif für eine digitale Umsetzung.



Brian Moriartys Perlenspiel reduziert die Spielelemente absichtlich auf das Wesentliche, damit sich seine Studenten auf den Kern der Spielidee konzentrieren.

von Infocom- und SCUMM-Adventures, für seine Studenten entwickelte, nennt sich „Perlenspiel“. Es besteht aus simplen Werkzeugen, mit denen sich einfache Puzzles bis zu komplexen Brett- und Strategiespielen umsetzen lassen. Moriarty verzichtet bewusst auf 3D-Grafik, Zwischensequenzen und ähnlichen Firlefanz und macht seinen Studenten sogar noch innerhalb des Perlenspiels Vorgaben, etwa dass sie mit einem Feld von 16 × 16 Pixeln auskommen müssen oder keinerlei Texterklärungen geben dürfen. Auf seiner Webseite finden Sie nicht nur die Open-Source-Engine als kostenlosen Download, sondern auch englischsprachige Lernmaterialien und im Browser spielbare Beispiele, mit denen Sie Ihre Spieldesign-Künste schärfen können.

Noch in diesem Jahr will Microsoft mit der Beta-Phase von Project Spark starten, einem spielerisch zu bedienenden Editor für Windows 8 und Xbox One.

Schnell und Schmutzig

Um Quick & Dirty zu testen, wie ein Spiel auf einem Monitor funktionieren könnte, müssen Sie sich nicht gleich in Entwicklungsumgebungen wie Unity, die Unreal- oder Cry-Engine einarbeiten. Dazu reicht oftmals ein Editor, wie ihn größere Spiele mitbringen. Geht es um ein rundenbasiertes Strategiespiel, bietet sich beispielsweise der auf der Script-Sprache Lua basierende Editor von Civilization IV an, mit dem man völlig neue Szenarien, Einheiten und Regeln aufstellen kann, sodass das Ergebnis völlig anders aussieht als Sid Meiers Original. Für ein Echtzeitstrategiespiel bietet sich eventuell der Editor von Warcraft 3 an, mit dem schon die ersten Versionen von „Defense of the Ancients“ (DotA) entstanden sind, bevor Valve daraus ein komplett eigenes Online-Spiel machte.

Eine Geschichte und Puzzles für ein Adventure lassen sich beispielsweise im kostenlosen Adventure Game Studio ausarbeiten, für ein Rollenspiel eignet sich der für 30 US-Dollar erhältliche RPG Maker XP, für ein Arcade-Spiel oder Jump & Run die The Games Factory 2, die das britische Click Team für 49 Euro unter Windows vertreibt.

Garage Games bietet mit Torque 2D und Torque 3D zwei Open-Source-Umgebungen für 2D- und 3D-Spiele an, die sowohl Microsofts Visual Studio als auch Apples XCode unterstützen. Dazu verkaufen die Macher Addon-Packs, die Figuren, Umgebungen und Texturen für verschiedene Genres bereithalten. Englischsprachige Dokumentationen und Tutorials findet man ebenfalls auf der Webseite.

Selbst für Konsolen finden sich entsprechende Editoren, die ohne eine Zeile Programmcode auskommen. So erlaubt „Little Big Planet 2“ auf der PS3, einfache Arcade-Spiele und Jump & Runs zu entwerfen. Mit dem „Project Spark“ arbeitet Microsoft derzeit an einer Weiterentwicklung seines Kodu Game Lab für Xbox One und Windows 8. Darin kann der Spieler mit wenigen Tastendrücken dreidimensionale Landschaften kreieren und Figuren und KI-Gegner hineinsetzen. Allerdings beschränken sich sowohl bei Project Spark als auch bei Little Big Planet 2 die Elemente auf das, was die Baukästen selbst mitbringen. Der Entwickler kann allenfalls eigene Sprachaufnahmen oder per Kinect aufgenommene Bewegungsmuster einbringen. Eine Beta-Phase für Windows 8 soll noch in diesem Jahr starten, eine Version für Xbox One im Frühjahr folgen.

Vom GameMaker zu Unity

Zwei Entwicklungsumgebungen, die sich in den letzten Jahren wachsender Beliebtheit in der Indie-Szene erfreuen, sind der GameMaker von Yoyo Games und die Unity-Entwicklungsumgebung. So landeten unlängst Jonatan „Cactus“ Söderström und Dennis Wedin mit dem Top-Down-Shooter „Hotline Miami“ einen auch kommerziell erfolgreichen Indie-Hit, den sie komplett im GameMaker umsetzten. Statt aufwendiger 3D-Grafiken kommt das Spiel mit relativ simplen 2D-Figuren aus und macht sein technisches Manko durch seinen grafisch extrovertierten Stil, die aufregende Musik und den hohen Schwierigkeitsgrad wieder wett. Einen ersten Eindruck des GameMakers vermittelt bereits die kostenlose Studio-Version für Windows und Mac OS X, mit der sich Prototypen mit einer begrenzten Anzahl von Objekten umsetzen lassen. Die Professional-Version für 100 US-Dollar bringt einen Texturen-Manager mit und unterstützt Team-Funktionen, sodass mehrere Entwickler am selben Projekt arbeiten können. Will man sein Spiel für Linux, Android, iOS oder HTML5 exportieren, lassen sich diese Module hinzukaufen, oder man greift zur Master Collection mit allem Drum und Dran für 800 US-Dollar.

Während der GameMaker sich vor allem auf 2D-Spiele versteht, hat sich im 3D-Bereich Unity etabliert. Zum einen können Entwickler im integrierten Asset-Store einzelne Module, die sie für ihr Spiel eventuell gebrauchen können, von anderen Entwicklern kaufen (oder ihre eigenen verkaufen), zum anderen unterstützt Unity von PC-Betriebssystemen über Mobilgeräte, Browser-Plug-ins bis zu Konsolen eine sehr breite Hardware-Basis. Programmiert wird hauptsächlich in C# und JavaScript. Wer sich hier einarbeitet, kann sich später auch bei größeren Entwicklerteams aus der Spieleindustrie bewerben.

Sie können ihre eigenen Spiele, die unter Windows, Mac OS X, Android, Blackberry, iOS oder mittels Plug-in im Browser laufen, mit der kostenlosen Unity-Version erstellen und sogar kommerziell vertreiben. Erst wenn Sie mit ihren Spielen mehr als 100 000 Dollar pro Jahr umsetzen, müssen Sie 1500 Dollar für die Pro-Version berappen, die zudem noch einige verbesserte Grafik-Effekte mitbringt. Selbst Konsolenspiele lassen sich per Unity umsetzen. Dazu benötigen Sie allerdings eine Developer-Konsole mit zugehörigem Software Development Kit von Microsoft, Sony oder Nintendo, die branchenüblich etwa das Zehnfache einer Retail-Konsole kostet.

Unity unterstützt in der Pro-Version inzwischen das Shader Modell 5.0, DirectX 11 und OpenGL ES 3.0, kann der populären Unreal Engine 3 grafisch also durchaus Paroli bieten. Epics Unreal Engine ist kostenlos, solange Sie ihr Spiel nicht kommerziell

vertreiben. Sie unterstützt neben Windows und Mac OS X auch iOS. Will man sein Spiel auf Android oder eine der aktuellen Konsolen portieren, benötigt man eine kostenpflichtige Full Source Lizenz, die einen Zugriff auf den C++-Code erlaubt. Preise muss man direkt bei Epic erfragen. Mit der Lizenzierung der Weiterentwicklung Unreal Engine 4 für Windows, PS4 und Xbox One hat Epic vor Kurzem erst begonnen.

Ähnlich verhält es sich mit der in Deutschland entwickelten CryEngine 3, die mit ihren besonders schnellen Renderern und aufwendigen Shader-Effekten wirbt. Für nicht kommerzielle Spiele und zu Ausbildungszwecken stellt Crytek die Engine kostenlos zur Verfügung und erlaubt sogar den direkten Zugriff auf den C++-Code. Allerdings läuft die kostenlose Entwicklungsumgebung nur unter Windows und spuckt auch nur Windows-Spiele aus. Kommerzielle Anbieter müssen 20 Prozent ihres Umsatzes an Crytek abtreten.

Plattformfrage

Doch so flexibel Unity auch ist, sollte man sich als Anfänger nicht täuschen lassen: Um die Möglichkeiten auszuschöpfen, braucht es viel Zeit, um sich in die 3D-Entwicklungsumgebung einzuarbeiten. Da kann es manchmal schlauer sein, sein Spiel zunächst nur für eine bestimmte Plattform zu programmieren oder sich an eine spezialisierte, übersichtlichere Umgebung zu halten.

Der GameMaker erfreut sich unter Indie-Entwicklern großer Beliebtheit, bringt er doch viele Annehmlichkeiten für 2D-Spiele mit und unterstützt viele Plattformen.

Auch wenn in den letzten Jahren verstärkt Browser-Spiele und Smartphones aufgekommen sind, ist Windows nach wie vor das am meisten verbreitete Betriebssystem. Wer sich hierfür entscheidet, hat bei den Entwicklungswerkzeugen die größte Auswahl, als Programmiersprache setzt ein Großteil neben C++ auf C#. Doch man sollte sich eine Portierung auf andere Plattformen möglichst offen halten. Wer beispielsweise für die Grafikausgabe einen eigenen Renderer für die DirectX-Schnittstelle programmiert, kann ihn mit Ausnahme der Xbox-Konsolen und der Windows Phones nicht auf anderen Plattformen nutzen. Linux und Mac OS X wie auch die übrigen Konsolen unterstützen hingegen OpenGL, dessen abgespeckte Variante OpenGL ES auf mobilen Geräten mit Android und iOS zu Hause ist. Zudem hat Microsoft die Entwicklung seiner XNA-Umgebung mittlerweile offiziell eingestellt, wie auch die Zukunft von DirectX unklar ist. Wer die Spieleprogrammierung lernen will, sollte sich deshalb besser anderweitig umsehen.

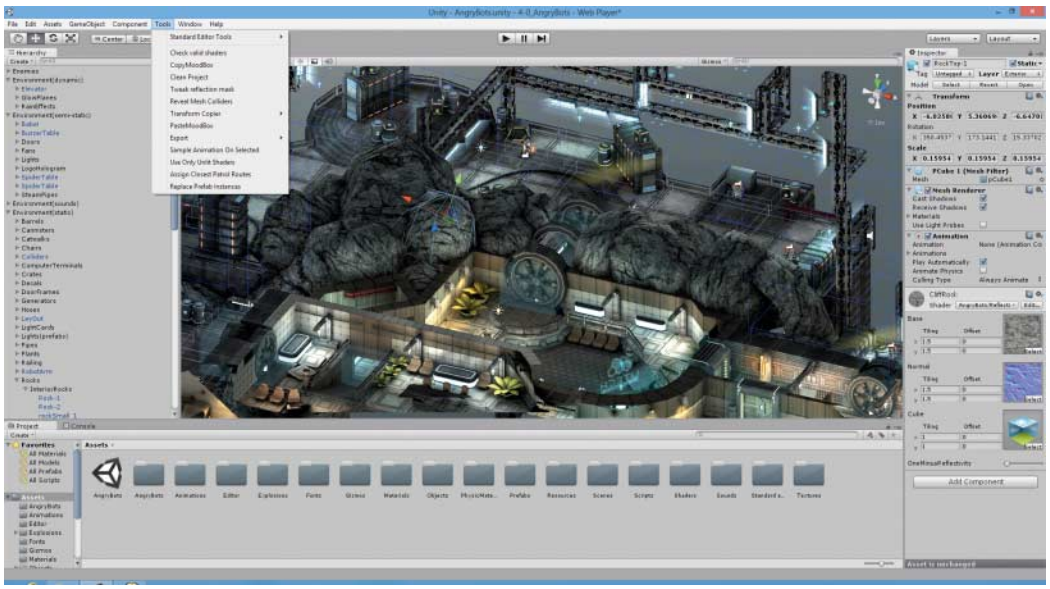
Linux hat sicherlich den Vorteil, dass man dort nicht nur das Betriebssystem, sondern auch viele Entwicklungstools kostenlos bekommt. Dieser Vorteil birgt jedoch auch einen gravierenden Nachteil: Weil Linux-Nutzer nur selten Geld für Software ausgeben, hat man es als Entwickler deutlich schwerer, mit Linux-Spielen Geld zu verdienen. Ändern könnte sich dies in den kommenden Jahren mit der Vertriebsplattform Steam von Valve und dem angekündigten Steam OS. Doch Verkaufsplätze auf dieser Plattform sind heiß begehrt, sodass man als Einsteiger nicht darauf spekulieren sollte, dort unterzukommen. Größere Chancen hat man schon bei dem Alternativ-Vertrieb von Desura.

Mac OS X hat etwa einen mit Linux vergleichbaren Marktanteil und (aufgrund der höheren Hardware-Preise) durchaus solvente Anwender. Hier programmiert man größtenteils in Objective C. Apple stellt mit XCode eine Entwicklungsplattform kostenlos zur Verfügung sowie den App Store als Vertriebsplattform, auf der Hunderttausende Programme um die Aufmerksamkeit der Nutzer buhlen. Steam ist hier ebenfalls präsent, Desura noch nicht.

Ein Mac-Rechner mit XCode ist auch die Voraussetzung, wenn Sie Apps für Apples mobiles Betriebssystem iOS programmieren möchten. Bereits ein kleiner Mac mini für 600 Euro reicht dazu aus. Im Vergleich zu Android macht es Apple Entwicklern mit einer wesentlich homogeneren Umgebung leichter, was die Anpassung an verschiedene Hardware und Betriebssystemversionen angeht.

Populäre iOS-Apps setzen meist auf 2D-Grafik, da diese über einen Touchscreen oftmals einfacher zu handhaben ist als eine 3D-Umgebung. Erste Wahl ist hier die kostenlose Entwicklungsumgebung Cocos2D, mit der selbst Hits wie Trainyard umgesetzt wurden. Cocos2D setzt auf Objektive C und XCode auf und macht es Programmierern vergleichsweise leicht, ein 2D-Spiel für das iPad oder iPhone umzusetzen. Zusätzlich zur kostenlosen Entwicklungsumgebung empfiehlt sich das englischsprachige Buch „Learning Cocos2D“ von Rod Strougo und Ray Wenderlich, die auf knapp 600 Seiten sämtliche Aspekte der Spiel-Entwicklung bis zur Veröffentlichung im App Store detailliert und anschaulich an einem Spieleprojekt durchgehen.

Viele Indie-Entwickler fangen der einheitlicheren Umgebung wegen zunächst mit einem App-Spiel für iOS an und setzen von



Will man ein 3D-Spiel auf möglichst vielen Plattformen inklusive Mobilgeräten und Konsolen veröffentlichen, kommt man kaum an Unity vorbei, zumal die Grundversion kostenlos ist.

dort aus ihre erfolgreichsten Titel auf Android um, das aufgrund der Vielzahl verschiedene Geräte und der vorhandenen älteren Betriebssystemversionen einen wesentlich größeren Test- und Support-Aufwand erfordert. Das erforderliche SDK gibt Google kostenlos auch für Windows-Rechner heraus, als Programmierspache setzen Android-Apps hauptsächlich auf Java. Eine Liste von Spiele-Engines, die auch Android unterstützen, finden Sie unter dem Link am Ende des Artikels.

Konsolen

In der vergangenen Konsolengeneration konnten erstmals auch unabhängige Entwickler ihre Spiele als Download-Version veröffentlichen. Microsoft hatte dafür einen eigenen Indie-Bereich auf der Xbox 360 eingerichtet, der Spiele aus dem XNA Developer Club beherbergte. Mit dem Tod von XNA und der bevorstehenden Veröffentlichung der Xbox One steht Microsoft jedoch vor einem Umbruch.

Für die Xbox One hat Microsoft sein neues Programm „Independent Developer at Xbox" (ID@Xbox) aus der Taufe gehoben. Allerdings wendet sich Microsoft zunächst an etablierte Studios, die bereits erfolgreiche Spiele vorzuweisen haben; die Initiative ist also nichts für Einsteiger.

Sony hat für Hobby-Programmierer sein Entwicklungsprogramm Playstation Mobile ins Leben gerufen. Die Programme laufen auf verschiedenen Android-Geräten von Sony und auf der Playstation Vita. Entwickler müssen keine Developer-Version der Geräte kaufen, sondern können mit Sonys SDK auf einem ganz gewöhnlichen Windows-PC programmieren. Dazu schreiben sie ihre Programme in C#, eine Anbindung an Unity ist geplant. Um die Spiele über den Playstation Store zu vertreiben, zahlt man pro Jahr eine Lizenz-Gebühr von 80 Euro.

Als dritter Hersteller erlaubt Nintendo unabhängigen Entwicklern, Spiele auf der Wii U zu veröffentlichen. Als Entwicklungsumgebungen kommen sowohl HTML5/JavaScript für Nintendos Web-Framework als auch Unity in Frage. Allerdings müssen Entwickler auch hier zunächst eine Developer-Version der Wii U von Nintendo erwerben. Mit im Paket ist bereits eine spezielle Version von Unity für die Wii U, die den hohen Kaufpreis der Developer-Konsole relativiert.

Abseits der drei großen Konsolenhersteller hat in den letzten Monaten das Start-up Oculus VR mit seiner Virtual-Reality-Brille Rift für Furore gesorgt. Bislang verschickt die US-Firma nur Entwickler-Versionen der Brille, die sich über einen DVI/HDMI-Anschluss an jedem PC oder Mac als zweiter Monitor betreiben lassen. Mit dabei ist ein Software Development Kit für Windows, Mac OS X und Linux, das sowohl Unity als auch das Unreal Development Kit unterstützt. Da die Kosten des Systems mit 300 US-Dollar überaus niedrig sind, bekommen Sie hier die Chance, eine völlig neue Art Videospiel zu entwickeln, bis die finale Version der Brille im Herbst 2014 erscheint.

Inzwischen locken selbst Konsolenhersteller Indie-Entwickler auf ihre Plattformen. Nintendos Wii U lässt sich beispielsweise auch per HTML5 oder in Unity programmieren. Die dazu nötigen Development Kits sind allerdings teuer.

Hits wie Trainyard wurden für iOS mit der kostenlosen Entwicklungsumgebung Cocos2D entwickelt.

Online

Aber Sie müssen sich ja nicht gleich festlegen, sondern können auch mit einem Browser-Spiel unabhängig vom Betriebssystem bleiben. Dazu programmieren Sie ihr Spiel wie unseren auf den folgenden Seiten vorgestellten Pac-Man-Klon in HTML5 und JavaScript. Schwierig wird es, wenn man die nur online verfügbaren HTML5-Spiele in eine Offline-Version verwandeln will. Dazu gibt es zwar passende Wrapper, die alle nötigen Laufzeitbibliotheken mit dazu packen, dadurch blähen sich die Programme aber stark auf. Wesentlich eleganter ist es, die Spiele direkt auf die Plattformen umzusetzen und deren Besonderheiten – etwa eine Touchscreen-Eingabe – im Spiel-Design zu berücksichtigen.

Alternativ laufen auch Unity-Spiele im Browser. Diese benötigen jedoch ein Plug-in, das derzeit nur unter Windows und Mac OS X verfügbar ist. Adobes Flash befindet sich derweil auf dem Rückzug, weil es von den meisten mobilen Browsern wie auch vom Internet Explorer unter Windows 8 nicht mehr unterstützt wird.

Frisch ans Werk

Wir konnten hier nur einen Einblick in die vielen Möglichkeiten und unterschiedlichen Werkzeuge geben. Am besten springen Sie gleich ins kalte Wasser und fangen mit einem konkreten Beispiel auf den kommenden Seiten an. Wenn Sie dessen einzelne Programmteile nachvollziehen, bekommen Sie einen ungefähren Überblick, was es heißt, ein Spiel zu programmieren. Weitere Inspirationsquellen finden Sie in unserer Rubrik „Freeware- und Indie-Tipps" ab Seite 190. In jedem Fall hilft eine Vernetzung mit Gleichgesinnten, die man beispielsweise auf dem Festival Ludum Dare oder der Game Developers Conference trifft. Alle Links zu den genannten Webseiten und Engines finden Sie unter dem Link am Ende des Artikels. Weitere Tipps und Tutorials für Spieleprogrammierer halten Web-Seiten wie spielprogrammierer.de, making-games.de, gamedev.net oder gpwiki.org parat. (hag)

www.ct.de/1323112 c't