

Experimental Coding for the OLPC

Projektbericht

Björn Birkenhauer, Max Wüsthube

07.05.2010

Dieser Projektbericht soll einen nicht-technischen Einblick in unser dreiwöchiges Praktikum am Lehrstuhl für angewandte Softwaretechnik der TU München geben. Ein weiterer Einblick in das Projekt ist unter <http://codingforolpc.blogspot.com/> möglich.

Ausgangssituation

Im Rahmen einer Kooperation zwischen der TU München und der Internatsschule Schloss Hansenberg wurde uns ein dreiwöchiges Praktikum in der Fakultät für Informatik angeboten. Nach weiterem Austausch von Interessen stand im Vorfeld fest, dass wir die drei Wochen am Lehrstuhl für angewandte Softwaretechnik von Professor Brügge verbringen dürfen. Unser Informationsstand bei der Anreise war, dass wir eine App für das iPhone entwickeln würden.

Entscheidung

Nachdem wir mit den aktuellen Projekten des Lehrstuhls bekannt gemacht worden waren, lag es an uns, eine Entscheidung zu treffen, welches Projekt wir bearbeiten wollen. Entgegen unserem vorherigen Informationsstand waren wir nicht fest zur Entwicklung einer App eingeplant, da es sich bei dem iPhone-Projekt, was zu dem Zeitpunkt anlief, um ein viermonatiges Praktikum für Studenten handelte, wir also nur eine geringe zeitliche Überschneidung hätten. Zur Auswahl stand also zum Einen die Entwicklung einer App für das iPhone/iPad unabhängig vom laufenden Studenten-Praktikum und zum Anderen die Entwicklung eines Programms für das „One Laptop Per Child“-Programm¹ (OLPC), dessen Ziel es ist, Kindern in Schwellen- und Entwicklungsländern günstige Laptops zum Lernen bereitzustellen. Der angestrebte Preis sind etwa 100\$, weshalb der Laptop auch 100-\$-Laptop genannt wird.

Die Entwicklung einer App bietet mehrere Vorzüge. iPhone und iPad bieten mit Touchscreen und Beschleunigungssensoren Eingabemöglichkeiten, die ganz andere Interaktionsmöglichkeiten bieten als alle anderen auf dem Markt befindlichen Geräte. Zudem sind beide Geräte sehr gefragt und wir hätten die Möglichkeit, unser Programm, so es denn gut wäre, in einem späten Stadium vielleicht sogar im AppStore zum Verkauf anzubieten. Problematisch bei der Entwicklung war allerdings, dass die Entwicklungsumgebung XCode nur auf Mac-Betriebssystemen läuft. Dementsprechend hätten wir nach derzeitigem Stand keinerlei Möglichkeit gehabt, unser Projekt nach den drei Wochen an der TU München weiterzuführen.

Beim OLPC verhält es sich bezüglich dieses Aspekts günstiger. Da Programme für den OLPC im Wesentlichen in Python geschrieben werden, können wir problemlos unter Windows entwickeln. Außerdem ist es natürlich sehr reizvoll, an einem so großen Open-Source-Projekt wie der Software Entwicklung für den OLPC mitzuarbeiten. Die Motivation, schlussendlich vielleicht sogar jemandem helfen zu können, ist natürlich ebenfalls nicht zu vernachlässigen.

Die Möglichkeit, nach unserem Praktikum weiter zu entwickeln, hat einen sehr großen Teil dazu beigetragen, dass wir uns schließlich für die Entwicklung eines Programms auf dem OLPC entschieden haben. Dort musste dann noch die Entscheidung getroffen werden, ob wir mittels Etoys oder mit „echtem“ Code entwickeln wollen, was relativ schnell zu Ungunsten von Etoys entschieden war, da wir echten Code entwickeln wollten und nicht in einer Umgebung arbeiten, die eher für das erste Erlernen von Programmierung gedacht ist. Jetzt musste nur noch ein Projekt gefunden werden, das wir umsetzen wollen. Felix, der seine Bachelorarbeit über ein Framework für den OLPC schreibt, schlug uns vor, dass wir eine App, die im Rahmen eines Projekts ein Jahr zuvor entwickelt worden ist, für den OLPC umsetzen. Diese Idee hat bei uns großes Interesse gefunden, so dass wir uns dafür entschieden haben. Das entsprechende Programm heißt weMakeWords. Es geht dabei darum,

¹ <http://laptop.org/en/>

Kindern im Alter von vier bis acht Jahren beizubringen, dass sie zusammenarbeiten müssen, um Erfolg zu haben. Sie setzen chinesische Zeichen aus ihren Bestandteilen zusammen, sind aber dabei darauf angewiesen, diese Zeichen von einem anderen Mitspieler zugespielt zu bekommen, da sie alleine entweder sehr lange brauchen, bis sie alle Bestandteile erhalten haben oder es sogar gar nicht schaffen.

Verlauf

Am Anfang stand zunächst einmal die Frage, welche Programme wir benötigen, um für den OLPC Programme zu entwickeln und sie auch zu testen. Sehr hilfreich in diesem Zusammenhang und auch bei späteren Fragen war uns das OLPC Wiki², in dem die Community zahlreiche Anleitungen, Tipps und Programme bereitstellt, damit es einem etwas leichter wird, für den OLPC zu entwickeln.

Da der OLPC eine auf Linux basierende Oberfläche hat, mussten wir unter Windows eine Emulation aufsetzen, auf der dann das Betriebssystem des OLPC läuft. Dies alles konnten wir mit Hilfe eines Installationspakets aus dem OLPC-Wiki auf einmal erledigen. Nachdem die Emulation erfolgreich lief, war der nächste Schritt, einen Datei-Transfer in die Emulation zu bewerkstelligen. Dies bewerkstelligten wir nach einiger Recherche mit einer simulierten Netzwerkverbindung zwischen unserem Betriebssystem und der Emulation.

Nachdem wir nun grundsätzlich in der Lage waren, Dateien auf die OLPC Emulation aufzuspielen, beschäftigten wir uns mit der Struktur, die Programme für den OLPC (dort „activities“ genannt), besitzen müssen, um zu funktionieren. Es werden nicht einfach Python-Programme auf der Festplatte abgelegt, sondern es muss eine bestimmte Ordner- und Dateistruktur vorhanden sein, damit eine activity funktionieren kann. Wir haben bereits bestehende activities abgeändert, um die Auswirkungen davon auf der Simulation zu testen. Dabei bestand zunächst die Schwierigkeit, dass die activities nur als eine Datei mit der Endung .xo aufgespielt werden. Dieses uns unbekannte Dateiformat entpuppte sich nach einiger Zeit als eine dem .zip Format sehr ähnliche Packweise, sodass es möglich ist, die Datei einfach als .zip Datei zu behandeln.

In Kenntnis dieser für unser Projekt notwendigen Grundlagen, begannen wir Python-Tutorials zu bearbeiten und auf der Emulation einfachste Programme, wie ein „hello world“, auszuführen. Sehr bald wollten wir jedoch nicht mehr nur Tutorials bearbeiten, so dass wir uns zunächst mit den UML-Diagrammen und dem Objective-C-Code des iPhone Originals unseres Programms beschäftigten. Uns schien die Klassenstruktur jedoch sehr umständlich, so dass wir uns entschieden, unserem Programm eine eigene, wesentlich einfachere Struktur zu Grunde zu legen. Also entwarfen wir ein eigenes UML-Klassen-Diagramm auf dessen Grundlage wir arbeiten wollten.

Das eigentliche Programmieren der activity machte schnelle Fortschritte, so dass wir innerhalb von ca. 1 ½ Wochen eine Windows und eine OLPC Einzelspielerversion erstellen konnten. Die wesentlichen Unterschiede dabei sind die nötige Anpassung der Datei- und Ordnerstruktur und eine Anpassung der Auflösung, was auch ein Bearbeiten der Bilder umfasst. Ein großes Augenmerk lag während der Programmierung darauf, die Größe der Dateien und die nötigen Berechnungen klein zu halten, um das Programm auf dem OLPC ausführbar zu halten.

² http://wiki.laptop.org/go/The_OLPC_Wiki

Ergebnis

Am Ende des Projekts stehen eine Windows- und eine OLPC-Einzelspieler-Version des weMakeWords Spiels, das ursprünglich für das iPhone programmiert wurde.

Wir als Schüler haben während des Projekts sehr viel gelernt. Während wir aus der Schule klare Zielvorgaben gewohnt waren, die sich, meist auf eine überschaubare Zeit beziehen, hatten wir an der TU München drei Wochen lang Zeit, komplett eigenverantwortlich an unserem Projekt zu arbeiten und dabei unsere eigenen Vorstellungen als Ziel zu nehmen. Wir mussten keine regelmäßigen Berichte vorlegen, wie es voran geht und wurden auch sonst nicht konkret auf unsere Leistung hin überprüft. Das eigenverantwortliche Arbeiten am eigenen Projekt war aber eine sehr spannende Erfahrung. Es ist das eigene Projekt, das man voran bringen möchte und das nur von einem selbst abhängt. Diese Motivation ist eine sehr positive Erfahrung. Doch natürlich bringt diese Selbstverantwortung auch Probleme mit sich. Wir haben uns gegen eine wahrscheinlich wohlüberlegte Klassenstruktur entschieden, mit dem Risiko, dass unsere eigene wesentlich schlechter funktionieren könnte. Doch wir haben das Risiko getragen und zum Ende hin auch gemerkt, welche Nachteile unsere Klassenstruktur hat, nämlich, dass ein unabhängiges Arbeiten an den Klassen zur gleichen Zeit quasi nicht möglich war. Dennoch haben wir ein Programm entwickelt, das unseren Zielsetzungen entspricht und die wesentlichen Funktionen des Originalprogramms imitiert.

Ausblick

Eine Fortsetzung der Entwicklung nach unserem Praktikum ist wünschenswert. Das nächste Feature, dessen Implementierung die OLPC Version des Spiels der iPhone-Version ein großes Stück näher bringen würde, ist die Einführung des Multiplayer-Modus, also eine Vernetzung von zwei bis vier OLPCs, deren Benutzer dann gleichzeitig spielen könnten und sich gegenseitig helfen. Derzeit ist nur ein Spiel mit vom Computer simulierten Mitspielern möglich.

Mit den im Verlauf unseres Arbeitsprozesses erzielten Ergebnissen wollen wir anderen helfen, selbst für den OLPC zu entwickeln. Dazu werden wir auf unserem Blog³ noch ein Tutorial bereitstellen, ein aktualisiertes UML-Diagramm erstellen und hoffentlich unseren Code zugänglich machen.

³ <http://codingforolpc.blogspot.com/>