A Contribution to Collaborative Learning Using iPads for School Children

Master's Thesis

Submitted to

Graz University of Technology

in Partial Fulfilment of the Requirements for the Degree of

Master of Science

by

Benedikt Kienleitner

Advisor

Assoc. Prof. PhD Martin Ebner

Graz, November 2013

Abstract

Collaboration has a positive effect on students' learning experiences as well as their social interactions. The intent of this master's thesis is to enhance the learning experience, stimulate communication and cooperative behaviour and thus improve learning. Making use of recent technological advancements (tablets) and gaming as a motivational factor, a prototype application in form of a multiplayer learning game for iPads was designed and developed. In a face-to-face setting, connecting up to four devices, the players (learners) have to solve word puzzles in a collaborative way.

Furthermore, a web-interface for teachers provides the possibility to create custom content as well as to receive feedback of the children's performance. A first field study at two primary schools in Graz showed promising results for the learning behaviour of school children.

Kurzfassung

Kollaboration hat einen positiven Einfluss auf die Lernerfahrung von SchülerInnen und deren soziale Interaktion. Ziel dieser Masterarbeit ist es, Kommunikation und Kooperation durch den Einsatz neuer Technologien zu fördern und im Zuge dessen die Lernerfahrung zu verbessern. Tablets bieten bisher noch nicht dagewesene Möglichkeiten kooperatives Lernen zu unterstütztenund Lerninhalte auf neue Art und Weise zu präsentieren. Im Rahmen dieser Arbeit wurde der Prototyp eines Mehrspieler-Lernspiels für iPads entwickelt. Das Spiel kann in Gruppen von bis zu vier Personen gespielt werden, die SchülerInnen sitzen dabei am selben Tisch. Ziel ist es, Worträtsel zu lösen. Kommunikation und Zusammenarbeit sind dabei die entscheidenden Faktoren.

Des Weiteren wurde eine Webseite für LehrerInnen entworfen, die es ermöglicht, eigene Inhalte zu erstellen und Rückmeldung über die Leistung der SchülerInnen zu erhalten. Eine erste Feldstudie in zwei Volkschulen in Graz lieferte vielversprechende Ergebnisse. Das Lernverhalten der SchülerInnen wurde positiv beeinflusst, das Spiel förderte sowohl Zusammenarbeit als auch Motivation.

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

…………………………….	……………..………………………………
date	(signature)

Acknowledgements

I would like to extend my sincere gratitude to my advisor, Assoc. Prof. Martin Ebner. He was always there when I had questions or needed support.

I would like to thank Behnam Taraghi for providing assistance during the development of my application. Furthermore, my gratitude goes to the two teachers of primary schools in Graz, Aureli Silvana and Angelika Kornberger, who made it possible to evaluate my app with their school children.

Last but not least, I would like to thank my family for supporting me during all those years of my studies.

Table of Contents

List of Figures

List of Tables

Source Code Listings

List of Abbreviations

API	Application Programming Interface
HTML	Hypertext Markup Language
ID	Identifier
IDE	Integrated Development Environment
iOS	iPhone Operating System (out-dated)
MVC	Model-View-Controller
PHP	Hypertext Preprocessor
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
TU	University of Technology
URL	Uniform Resource Locator
W3C	World Wide Web Consortium
XML	Extensible Markup Language

1 Introduction

The last decade has provided us with amazing new and innovative technological possibilities. New ways to deliver content and support learning have become available, whether on the software side, most notably Web 2.0 (O'Reilly, 2007), as well on the hardware side like the development of smartphones and tablets. In addition, computers, and tablets in particular, offer a perfect opportunity to present learning material in a more playful manner. Under the right circumstances, education with learning games can have many benefits (Mann, et al., 2002).

Educational games already exist in many forms, however, only a handful of games could be found that focus on the aspect of communication and cooperation among peers. It has been shown in numerous studies (Johnson, et al., 2009) that collaborative learning can have a positive effect on social behaviour as well as learning results and communication abilities (Jordan, et al., 2006) (Johnson, et al., 1999). Due to the fact that learning is an active part of the learner where knowledge and understanding is constructed by the learner (Holzinger, 2002), communication and collaboration are essential factors for this process. Learning is a highly social process and develops through conversation (Dewey, 1916) (Motschnig-Pitrik, et al., 2002).

1.1 Research Goal

Focusing on the aspect of cooperation and collaborative learning, the goal of this research is to use digital devices to connect learners to strongly assist the communication between peers. The fundamental idea is to develop an application where learners actively engage into collaborative work.

1.2 Overview of the Thesis

During the course of this master's thesis, a learning game for mobile devices was developed as well as a website for teachers to manage the learning content of the app. Furthermore, an evaluation of the project took place in form of a field study at two primary school classes in Graz.

The written part of this thesis is structured as following:

- *Chapter 2* provides the theoretical background of the thesis, discussing the three aspects that are the basis of the project: Collaborative learning, educational games and mobile devices in classrooms.
- *Chapter 3* will outline the idea and concept of the application.
- *Chapter 4* discusses the user-interface and features of the app, providing information on user interface design for mobile devices and designing user interfaces for children. Each screen of the app will be presented, outlining functionality and design.
- *Chapter 5* focuses on the implementation of the iPad application. Each class will be inspected, the technologies involved will be explained and significant pieces of code or functions will be illustrated.
- *Chapter 6* presents the website, the design and functionality as well as details regarding the implementation, the database schema and the interface to the user management system for schools.
- *Chapter 7* focuses on the evaluation of the app, outlining the setting of the field study and discussing results and outcomes.
- *Chapter 8* gives concluding remarks about the project, the evaluation and general thoughts on this field of research.

2 Theoretical Background

This chapter presents information on collaborative learning, educational games and the use of mobile devices in classrooms.

2.1 Collaborative Learning

Learning in schools can be divided into three kinds of interaction:

- Interaction between the teacher and the students.
- Interaction between students and the curriculum materials.
- Interaction among the students.

Instructional theory has long focussed on the first two types of interaction, while student-to-student interaction has largely been ignored. This is to some degree due to the fact that cooperative learning is not the easiest way to teach and is viewed with scepticism by some teachers. However, the way in which teachers structure student learning goals determines how students interact with each other. Those interactions are a major determinant of cognitive and affective outcomes of instruction (Smith, 1979) (Johnson, et al., 1991).

In simple terms, collaborative learning can be described as a situation in which two or more people attempt to learn something together (Dillenbourg, 1999). They will ask each other for information, evaluate one another's ideas and monitor one another's work (Chiu, 2000). These effects, however, do not automatically appear when students are placed together in groups. For cooperative learning to occur the groups and learning activities must be carefully chosen (Johnson, et al., 1991) (Laughlin, et al., 1991).

The benefits of collaborative learning are an increase in students’ engagement and their motivation to learn as well as a deeper understanding of learning material (Murphy, et al., 2009) (Prince, 2004). According to Vygotsky (Vygotsky, 1978), students who work together are capable of performing at a higher intellectual level than working individually. Furthermore, as a result of collaboration and communication among peers, an improvement of students' interpersonal relationships was noticed (Johnson, et al., 2009).

There are several ways to implement cooperative learning. A more formal approach would be to place students in groups and optionally assign a specific role for each student. The teacher provides the learning material and explains the tasks, monitors the learning activities of each group and intervenes to teach cooperative skills and assist in learning when needed. Finally, for each student and group, the results of learning are evaluated.

Another way is to form temporary, ad hoc groups that last for only one class period. Those activities can be used to further focus students’ attention on the learning material and ensure that students cognitively process the material being taught. This approach can be used any time, however, it is especially helpful when students seem to lose focus on the lecture (Johnson, et al., 1991).

2.2 Educational Games

Information technology has changed the way we work, live, learn and entertain ourselves. A new generation of students has emerged, their learning preferences tend toward teamwork, experimental activities, structure and the use of technology (Raines, 2002).

Outside school, computer games have become an integral part of young people’s live, holding a special fascination and provoking a deep sense of engagement (Facer, 2003). Today, games are part of growing up, the average 8th grader plays video games for approximately 5 hours a week (Oblinger, 2004). A survey performed in 2003 stated that 69% of respondents had been playing computer games since primary school (Jones, 2003). Observing this trend, the following question arises: How can this new form of media be used for educational purposes?

Games can constitute powerful learning environments for a number of reasons (Oblinger, 2004):

- They can support multi-sensory, active, problem-based learning.
- Games favour activation of prior knowledge. Players must use previously gathered information in order to advance.
- Games require the transfer of learning from other venues of life. Being able to see the connection and transfer it to a unique situation is part of gameplay.
- They provide immediate feedback, enabling players to learn from their actions.
- They offer opportunities for self-assessment through the mechanism of scoring and reaching different levels.
- Games are often social environments. They can be played with others and involve large communities.

In addition, games can have a great influence on motivation. Learning in form of a game is believed to be more learner-centered, easier, more enjoyable, interesting and thus more effective (Kafai, 2001) (Papastergiou, 2009). Malone (Malone, 1980) pointed out that learning can be improved through using the three crucial factors: curiosity, fantasy and challenge.

Today there are numerous research studies carrying out the idea of using games for learning (Zechner & Ebner, 2011) (Hannak, et al., 2012) (Ebner, et al., 2011). To complete this section, two of those studies regarding learning games will be briefly presented. The first game was developed at Graz University of Technology.

iGeo is an online learning game and designed for the subject geography in secondary schools. The goal of the game is to spot locations, including capitals, mountains and important cities. Geographical facts are supplied along the way and a high score list with nicknames should further motivate players (Ebner & Holzinger, 2007).

The second game was developed in Sweden and is divided into several modules that contain highly interactive content with 60-80 hours of active learning. Subjects such as English, mathematics, physics and business administration are set in a learning environment where the user can explore, experiment and practice. For example, the student encounters a number of characters in the game that need the learner's help to solve tasks. A task could be to coach the career of a young journalist by using statistics or to help out elderly people that need health care (Oblinger, 2004).

In both cases, students found the learning experience more engaging and enjoyable. Furthermore, better learning results were achieved.

2.3 Mobile Devices in Classrooms

Mobile devices have become an integral part of our lives. In the first quarter of 2013 more smartphones have been sold than normal mobile phones[1]. Students are already using personal devices for learning outside of school. This raises the question: Will students who come to expect mobile personal devices outside of school demand to use them within school? (Chan, et al., 2006)

There are two key factors for the use of mobile devices in classrooms (Liang, et al., 2005):

- One device per student.
- A communication network that supports peer-to-peer connections and/or internet connectivity.

In such an environment, handheld devices can have numerous educational benefits, a few of them are listed below (Kloper, et al., 2002):

- *Portability*. The ease of movement with the device creates learning environments that have not been possible before.
- *Social interactivity* through wireless communication. Peer-to-peer communication makes data exchange, face-to-face interaction and collaboration possible.
- *Customization* to the individual's path of navigation.

[1] http://www.iphone-news.org/2013/04/26/smartphones-vs-handys-erstmals-mehr-smartphones-verkauft-apple-mit-17-3-prozent-marktanteil-48678/ (Last visited: Nov. 2013)

- *Connectivity*, establishing a shared environment for data collection among distributed devices.
- *Combination of digital and physical worlds.*

Educational applications on mobile devices can be categorized in three main types (Pinkwart, et al., 2003) (Chan, et al., 2006):

1. An interface to a desktop program to extend the use of a desktop application.
2. A stand-alone application running on the mobile device allowing cooperation via direct communication between the devices.
3. As a portal to a shared space that resides on a server.

In recent years there has been a trend towards an integral use of handheld devices in classrooms instead of the occasional visit of computer labs. Portability and peer-to-peer connectivity make them a perfect choice to assist cooperative learning approaches. Thus, the collaboration of students through different mobile devices has become an import research issue. Studies demonstrated that mobile technology can aid or actively support collaboration (Zurita & Nussbaum, 2004).

However, technological advancement is usually faster than its adoption, especially regarding technology for educational purposes in schools. According to Rogers (Rogers, 1995), there will always be a group of innovators who actively contribute to the use of new technology followed by technology enthusiasts and an early majority of pragmatic users. On the opposite side stands the late majority, people who are generally against innovations and a group of people that do not accept innovation at all. A study performed in 2006 has tried to roughly estimate this behaviour regarding the acceptance of mobile in classrooms.

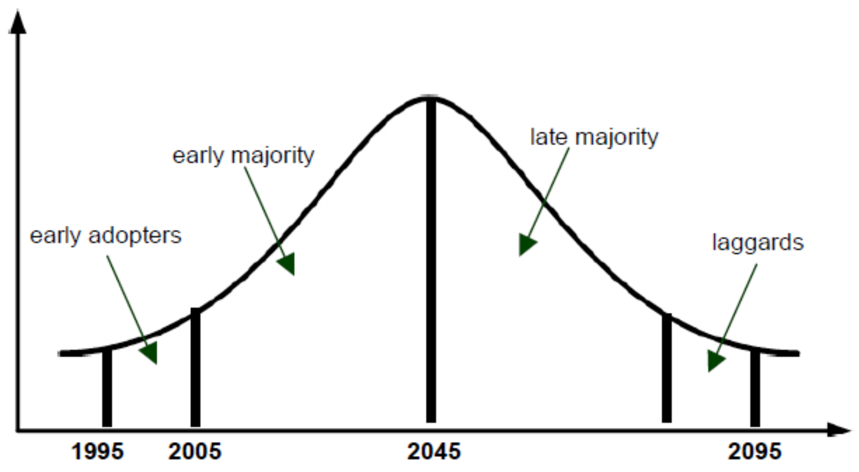


Figure 1 Acceptance of Mobile Devices (Moore, 1999) adapted by (Chan, et al., 2006)

According to figure 1, there will be an upsurge of changes in education during the next two or three decades. Nevertheless, the process of adopting mobile devices or similar technologies for the use in classrooms will most likely not happen in the matter of a few years.

Perez (Perez, 2002) offers another point of view on that matter. In his reasoning, the introduction and incorporation of a new technology is divided into two phases. During the installation phase, there will be a time of explosive growth which will be followed by a period of disappointment and confusion. Finally, a turning point will be reached which will start a period of rethinking and rerouting the development leading to the deployment period, where the technology has reached a certain maturity. Regarding the adoption of mobile devices in classrooms, the development is still in its very early stages, and a lot of time and research will have to be performed until a certain point of maturity is reached. Still, the technology and its use for education holds much promise and in the years to come a lot of interesting changes and opportunities will present themselves.

An example would be the development regarding mobile devices and their use for educational purposes in developing countries. There has been a rapid growth of mobile phone access during the last years in Sub Saharan Africa, especially among the youth of South Africa. Ghana is one of the states with the highest mobile penetration rate. Up to 90% (Dawes, 2011) of young people have access to a SIM card or mobile device. In contrast, other technological options that might deliver educational content are practically not available. Only 9% of Ghana's population have access to PC's or laptops at home, while the ratio of mobile cellular subscriptions in general is at 74% (ITU, 2011).

The lack of learning equipment as well as trained teachers is one of the biggest problems. Mobile based solutions can help to compensate these issues. M-Learning has the potential to "reach people who live in remote locations where there are no schools, teachers, or libraries" (Ally, 2009) (Grimus, et al., 2013).

While a holistic, sustainable approach to m-Learning in Ghana has yet to be found, the chances of a large scale adoption of m-Learning in Africa are discussed around the world. M-learning seems to be an ideal solution to cope with the needs of education in developing countries (Grimus, et al., 2013) (Grimus & Ebner, 2013).

During the course of this chapter, the benefits of mobile devices and other innovative technologies have been pointed out on several occasions. However, as a final remark, the risks and side effects that come along with them should be noted as well (Chan, et al., 2006):

- Privacy issues and data security.
- The blending of informal and formal environments. This phenomenon can already be observed at the working place when the line of formal and informal work is blurred, leading to an unbalanced living situation.
- A certain dependency on the industry. More advanced devices and software will always be more expensive and thus out of reach of economically less developed communities.
- Environmental and ecological costs.
- Learning devices that are used to learn socially destructive contents.

Most of these issues are not directly linked to the use of handheld devices in classrooms, but an inherent part of using computers.

3 Idea and Concept

TU Graz has been developing learning applications for iOS for several years, already starting in 2010. A number of workshops have been given on app design and how to achieve users' satisfaction (Ebner, et al., 2010). In addition, several studies have provided valuable insights in the design of applications for iPads in school classes. Two points that should be improved stood out above the others (Huber & Ebner, 2013):

- *Missing feedback for teachers.* Mobile applications are usually standalone programs and learning takes place as an interaction between the student and the application, excluding the teacher from the process. There is no possibility for the teacher to get insight into a pupil's performance, how many exercises are done or if the learning goals are achieved.
- *Lack of collaboration.* Mobile devices provide unprecedented possibilities to enable and assist in cooperation and collaboration. While current learning applications are able to enhance individualised learning, developers of mobile learning apps seem to have taken no special interest in this area.

As far as it could be determined only a handful of applications are available in the App Store that actively support collaborative learning, even fewer cooperative learning games and apps that have been specifically developed for the use in classrooms.

However, a learning game called "MatheBingo[2]" with similar focus on collaboration has been developed at the IICM[3] previously to this work. Valuable insights on technical aspects of the implementation as well as decisions regarding the design of the app could be gathered from the project.

[2] https://itunes.apple.com/at/app/mathebingo/id568942997?mt=8 (Last visited: Nov. 2013)
[3] Institute of Information Systems and Computer Media (Graz University of Technology)

Building on these experiences, the fundamental idea of this project was to develop an application where learners actively engage into collaborative work and to combine this with the benefits inherent to games. Furthermore, the app should provide an interface for teachers, so that they would be able to receive feedback of the pupils' performance and to create their own learning content fitting to the curricula of the class.

3.1 Methodology

The practical part of this master's thesis consists of two steps. The development of a prototype application and website, and a field study to test and evaluate the prototype. The application is a cooperative multiplayer learning game for iPads, the website a user interface for teachers to create custom content and to monitor the children's performance.

The field study took place at two primary schools in Graz. In a proper classroom setting, the app was tested in form of a participatory observation. Afterwards, interviews with a selected number of children were performed.

During the course of the next two chapters, an extensive description of the iPad application will be provided, starting by further illustrating the concept of the app, including a short explanation of the game mechanism. This will be followed by a detailed description of the design and features, including technical background on user-interface design for children and games. The next chapter will focus on the implementation. The technologies and frameworks involved will be outlined as well as the system's architecture and class design, followed by a detailed description of the implementation.

3.2 Idea and Concept

"Buchstaben Post[4]" is a learning game for schoolchildren between grades 1 to 4. The aim is to teach children the correct spelling of words with a focus on collaboration and communication between the players.

[4] https://itunes.apple.com/de/app/id736836885?mt=8 (Last visited: Nov. 2013)

The idea is the following: Connect up to four devices in a peer to peer session. While it is still possible to play alone, it is intended as a multiplayer game with 2 to 4 people. To achieve a level of cooperation, all players have to work together to progress to the next round. The teammates are supposed to sit on the same table. The tablets are placed beside or in front of each other, resembling the setting of a classical board game. The following graphic is an abstract representation of such a game setting with four players, respectively four iPads, lying in front of each other on the same table. All players are able to see the content displayed on all tablets, so they are able to help each other out in several ways. They can either help by discussing the solution or actively take action by reaching out and interacting with the iPad of a teammate. This is a unique setting, establishing a level of social interaction and the ability to work together that has not been possible before with digital devices. It is a great display of the advantages of tablets or mobile devices over conventional personal computers or even notebooks.

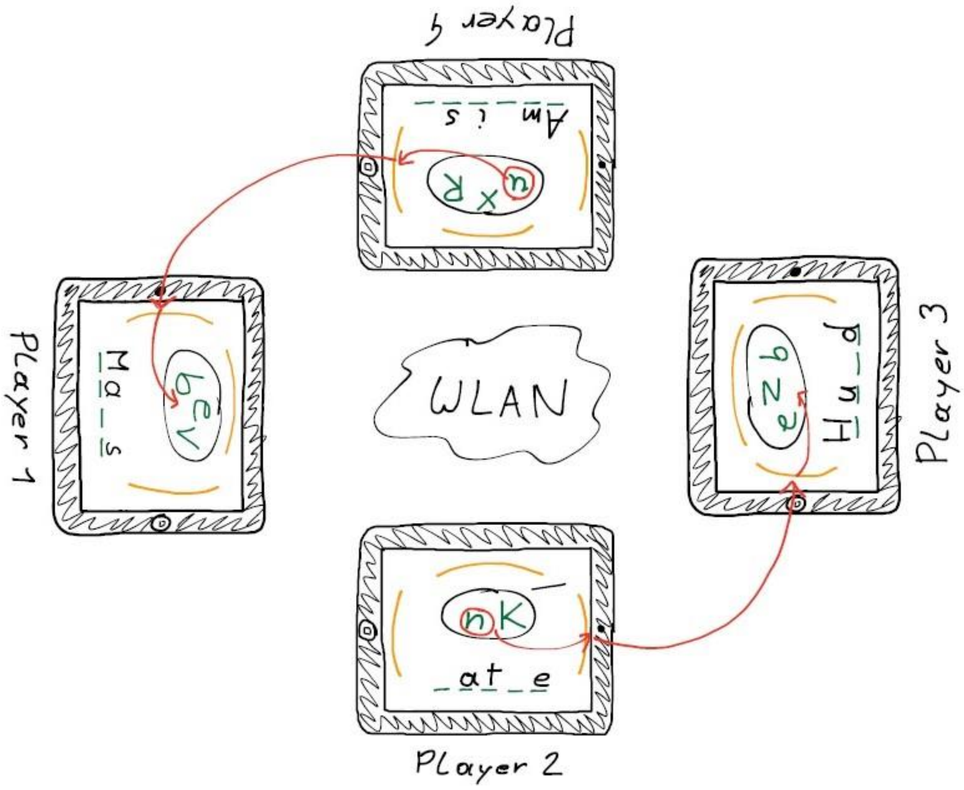


Figure 2 Game Setting

Another significant feature of the application is the following:

Many current learning games are too general regarding their content and lack a connection to curricula in school (Egenfeldt-Nielson, 2005). Thus, another important aspect is to provide teachers with the opportunity to create custom content for their children and to receive feedback. Especially school children in the first grades are only able to read or write a limited number of different words, so the difficulty of the game had to be adjustable to fit the curriculum of the school classes. For this purpose a user interface for teachers in form of a website had to be designed. The website will be outlined in chapter 6.

4 User Interface and Features

A short description of the features and game mechanism of the app will be followed by some theoretical background on user interface design. Furthermore, a detailed analysis of every screen of the app will be provided with regard to design and functionality.

4.1 Short Introduction of the Game

As mentioned before, the game can be played in single player mode. If that's the case, the goal of the game is simply to guess the correct spelling of a word. A hint in form of a sentence or a question and a number of letters are provided. The player has to substitute the missing letters of the word to guess. This is a well-known concept which already exists in numerous applications. The idea is to use that approach and expand it in a way that would create a multiplayer application, where players not only have to guess their own words, but help each other out as well.

The game is not competitive, which means that the players are not "punished" for choosing a wrong letter, they can try as many times as they want. The game will not progress to the next round until all players have finished their words. Since it is meant to be played in a face-to-face setting, all players should be able to see the words of their teammates. Children who finish faster are thus able to help others out.

In order to further reinforce the idea of a "common goal", the players are confronted with another challenge. While some players may possess all the missing letters of their word, other players will not be so lucky – they have to get the missing letters from one of their teammates.

The following sections provide a more detailed explanation of the game mechanism.

The application was originally developed for Austrian school classes. It is therefore only available in German language. However, the design of the app allows easy adaption to multilingual support in future versions. Apple provides a detailed manual[5] on how to internationalize an application. In this case, only the help messages and button names have to be placed in a .strings file, where Xcode stores localized string resources. Teachers are able to create their own wordlists through the web-interface in whatever language they choose.

The game is composed of four screens, not counting the splash (or loading) screen. The next section addresses user interface design in general, before explaining each screen and the corresponding features in detail.

4.2 User Interface Design

This section presents general guidelines and principles that were taken into consideration concerning the design of the game and especially in creating interfaces for children.

With the exception of a few elements, all graphics and animations were produced without the help of professional designers. Therefore, several aspects of user-interface design had to be studied carefully:

- User interface design for mobile devices and iPads in particular.
- The design of educational applications and games.
- Designing software for children.

The correct design of the user interface is one of the most important parts of an application, in this case probably even the foremost one. Concerning games, graphics and usability are an essential factor. This is true for every application, and especially for mobile devices and iOS in particular, user demands are on a very high level. Design and usability are key factors to the success of a mobile application (Gong & Tarasewich, 2004). Users are not as lenient or patient with badly designed interfaces as they were a decade ago. While content and features are still important factors, they might not be considered at all because the app will be discarded after a

[5] https://developer.apple.com/library/ios/documentation/MacOSX/Conceptual/BPInternational/Articles/InternatSupport.html (Last visited: Nov. 2013)

few clicks, due to the fact that time is often critical to a mobile device user (Poupyrev, et al., 2002).

When designing graphics and interfaces for games as well as for children, factors apart from usability and functionality have to be considered. Aesthetics are part of an overall enjoyable user experience (Karlsson & Djabri, 2002). In addition, colour and its manipulation are an important aspect for visual interfaces (Schneiderman, 1998). In general, it can be said that emotion plays a large part in our interaction with objects (Norman, 2004).

These statements hold true for users of every age, however, more aspects have to be considered when designing interfaces for children. Children require emotional support and a feeling of success (Erikson, 1963). To achieve this, it is important to properly guide the children through the application to avoid frustration (Gossen, et al., 2012). In addition, positive feedback was provided whenever possible, primarily through the use of simple graphics and animations or sounds, making the interface more attractive for children as they prefer colourful designs with multimedia content (Large, et al., 2002) (Naidu, 2005) (Budiu & Nielson, 2010).

The app is intended for the use in primary schools. At this age, children are reading slowly and are still learning to write (Stuart, 2007). They will have limited domain knowledge (Hutschinson, et al., 2005) and difficulties with typing using a keyboard (Solomon, 1993). Those considerations had to be taken into account, especially regarding the design of the login and connection views.

Children’s fine motor skills are different than those of adults. Studies regarding pointing tasks have shown that children’s performance is below adults by several degrees. In most cases the target size had a significant effect on children’s accuracy (Hutschinson, et al., 2005). This is an import fact to consider. Due to the nature of the application, the app has more interactive content than usual. Beside buttons and menus, the whole game screen is interactive, and dragging the letters into right position requires precise movement.

Furthermore, children will have difficulties with thinking abstractly (Piaget, et al., 1969). Thus, menus and similar navigation elements should not contain abstract concepts and metaphors have to be carefully chosen. However, images better match children’s cognitive skills than written words (Hackfort, 2003). Therefore, whenever possible, icons, images or animations are used to indicate a certain functionality.

While the statements listed above hold true for every kind of application, be it a website, desktop program or handheld app, user interface design for mobile devices involves additional constraints and requirements. In the first place, every application developed for the iOS should adhere to Apples iOS Human Interface Guidelines[6]. However, those are of a more general nature and deal primarily with the correct use of navigation elements in a standard app and are not really suited for this software due to the unique interface of a children's game.

During the last years, numerous learning applications for mobile devices were developed at TU Graz, with a special lecture for designing apps for iOS. Also, a number of educational applications for children were developed, and several tests and studies have taken place at Austrian iPad classes were those apps were evaluated in a proper classroom setting. In addition, a number of workshops have been given on app design and how to achieve users' satisfaction (Ebner, et al., 2010). The insights gained from those experiences have been summarized in the iPad Human Interface Guidelines (Ebner, 2010), as well as the extended guidelines focusing on mobile learning (Huber & Ebner , 2013). Important points for this application are:

- *Aesthetic Integrity.* The look of an application should incorporate its function.
- *Consistency.* Consistency in the user interface.
- *Presentation* of the content in a beautiful, often realistic way.
- *Direct Manipulation.* Direct Control through multi-touch gestures.
- *User Control.* The user should be in charge of things.
- *Feedback.* Immediate feedback after operations.
- *Metaphors.* Relation of virtual objects to real world objects.
- *Start Instantly.* Display a launch image that closely resembles the first application screen.
- *Create custom icons.*

The following sections will focus on each screen of the application and explain the corresponding design, functionality and features.

[6] https://developer.apple.com/library/ios/documentation/userexperience/conceptual /mobilehig/ (Last visited: Nov. 2013)

4.3 Menu Screen

The menu screen is the first screen that is being displayed, immediately after the application has finished loading.

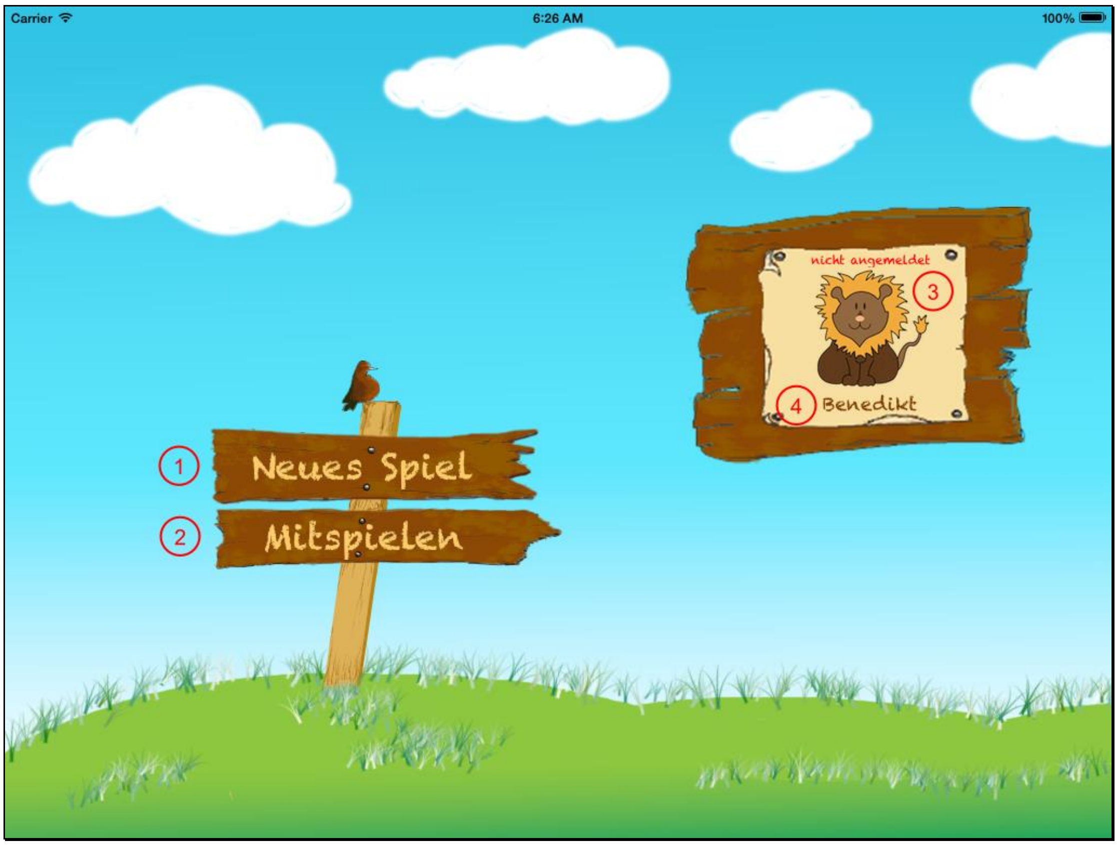
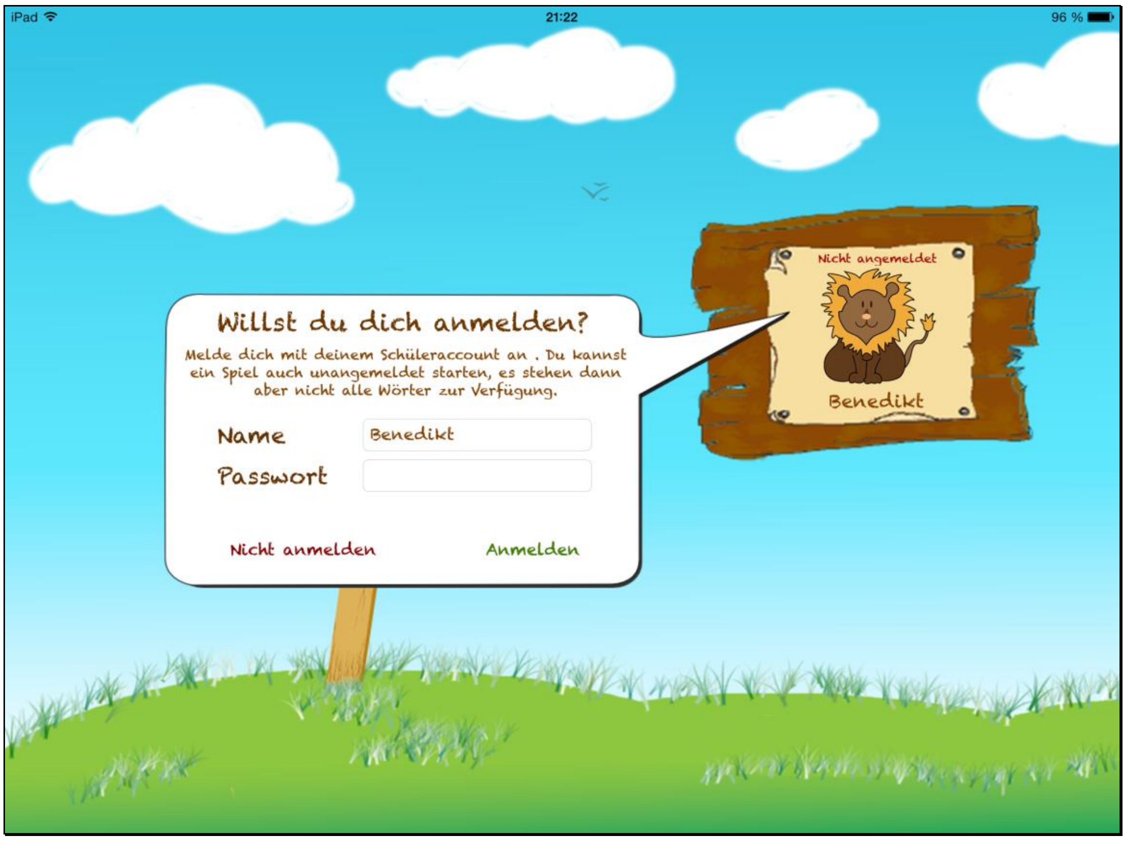


Figure 3 Menu Screen

To keep it as simple as possible, the game menu contains only two options. The player can either start a new game (1), or join an already existing one (2). A player icon can be selected by clicking on the corresponding picture (3). This will loop through all available icons. Below the icon, the username is displayed (4). A single touch on the name will open a form to login or change the username. Once changed, the name will be stored locally on the device. The text above the icon indicates the login status of the user. A simple fade in/out animation indicates the possibility to change the icon/username by touching the picture/name. In retrospect, this little bit of help was completely unnecessary, the children are obviously used to the concept of avatars or user icons, and it took them only a few seconds to figure out how to change their picture, very much to their delight. At this point it is worth mentioning that player icons hold a special fascination for children, assuming that they are designed according to children's imagination. As observed during the field tests, children love to search and identify themselves with an avatar. They can also be used as a great motivational factor, when they are granted for certain accomplishments in the game.

Returning to the game menu, once a player has chosen to start a new game or to join one, he/she will be asked to sign in. This opens a login form, as depicted in the figure below.



Willst du dich anmelden?

Melde dich mit deinem Schüleraccount an . Du kannst ein Spiel auch unangemeldet starten, es stehen dann aber nicht alle Wörter zur Verfügung.

Name: Benedikt

Passwort:

Figure 4 Menu Screen with Login View

To log into the game, an account at the TU Graz user management[7] system is required. While it is possible to play offline, the sign in is required for several reasons.

1. To download a custom list of words prepared by the teacher.
2. To store which words a player has already finished.

[7] The user management system will be explained in section 3.3.

3. To upload data regarding the children’s performance.

Those three points will be further discussed after illustrating the possibilities a player has when playing off- or online.

4.3.1 Differences between Off- and Online Play

Apart from technical issues, schools might restrict internet usage of pupils for several reasons. Thus, the app was designed to work offline as well as online. In case that only a limited number of devices are able to establish an internet connection, it is enough for the device which acts as a server (the one that started a game) to be connected in order for all players to benefit.

The following graphic illustrates the communication between the web API, game server and clients.

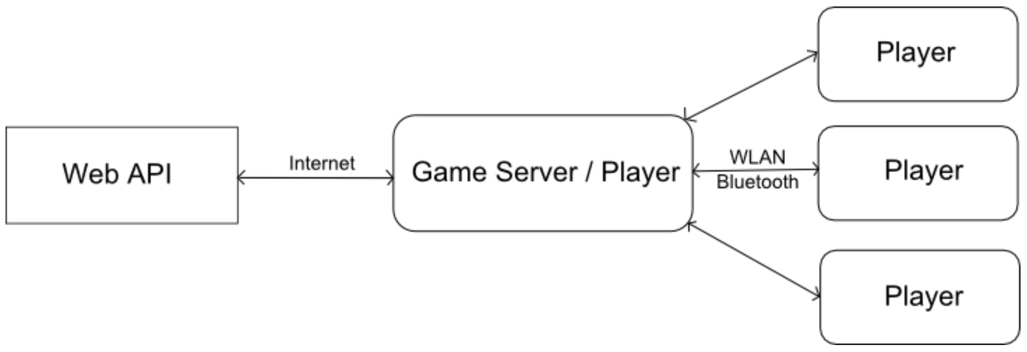


Figure 5 Communication between Web API, Server and Clients

While the connection itself is a peer-to-peer connection between all devices, the player who started the game takes on the role of server regarding communication with the web API. The game server is also responsible for the distribution of words and questions among the players. Other than that, it acts as a normal player.

For the sake of simplicity, a player will be referred to as “online” when he/she is signed in, although it is also possible to choose not to login when the device has an active internet connection. The following figure will further illustrate the different con-stellations in which playing is possible.

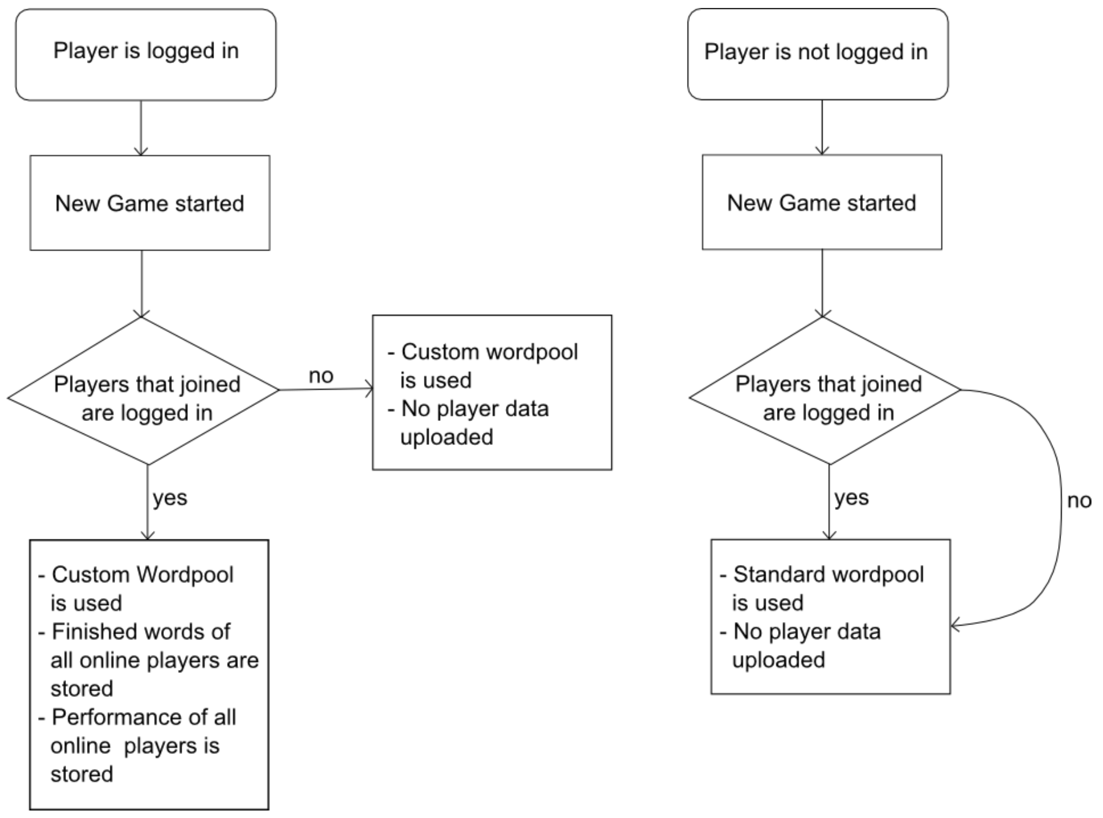

Player is logged in
New Game started
Players that joined are logged in
no
- Custom wordpool is used
- No player data uploaded
yes
- Custom Wordpool is used
- Finished words of all online players are stored
- Performance of all online players is stored
Player is not logged in
New Game started
Players that joined are logged in
yes
no
- Standard wordpool is used
- No player data uploaded

Figure 6 Flowcharts Off- Online Gaming

Basically, there are three different situations. When the player who started the game, hence referred to as game server or server, is not logged in (figure 6 on the right), there is no possibility to download a custom wordpool from the web-interface. For this case a small wordlist (around 60 word question pairs) was compiled, which is stored locally on the device. It makes no difference if other players of the same game are online or not, since all communication with the web-interface is relayed over the game server.

The preferred situation would be the one depicted in the left flowchart (figure 6), where all players are signed in. This way, they are able to play with a custom word-pool and it is possible to upload data of their activity. However, in the third scenario, only the game server needs to be online for all players to use the same custom wordpool, but user data is only stored from the players who are online.

4.3.2 Features of the Web-Interface

The design and implementation of the website and API will be discussed in chapter 6, this section focuses only on the main features of the web-interface regarding the functionality of the app. The features are the following:

1. *Download a custom list of words.*
   Teachers have the opportunity to create custom wordlists (a wordlist or wordpool contains pairs of words and questions/hints) for the children.

2. *Store which words a player has already finished.*
   Words are randomly drawn out of the assigned wordlist. However, words that have been selected once have a lower probability to appear again. Also, in order to ensure that newly added content will not be omitted, “fresh” words have a much higher chance of being drawn.

3. *Upload data of the children’s performance.*
   A learning application should offer feedback to the students as well as the teachers. While technically possible to store a multitude of data of the children’s activity, it was hard to gather useful information for this particular setting. This is due to the fact that the game is highly cooperative and players are not able to complete a word without help of others. In the end, it was decided to log the number of completed words for every pupil and the time it took for all players to finish one round.

4.3.3 The Login Procedure

Difficulties with the input of login credentials lead to several insights. As mentioned earlier, school children are reading slowly, are still learning to write (Stuart, 2007) and will have difficulties with typing using a keyboard (Solomon, 1993). Nevertheless, there was no way to completely avoid the login procedure, since users have to be identified through some method. While the username is stored on the device so that it will remain after closing the app, the password has to be filled in whenever the app is launched. The children were using the devices on a regular basis, so it was assumed that they would be familiar with password input fields. Still, some had considerable difficulties in typing in the correct password. The problem was encountered not only when testing this application, but also with another similar learning app that required a login.

As far as it was possible to determine, the problem resulted from the password field’s function to blank out letters (or switch them with black dots). While the children were able to fill in the username without problems, the missing visual feedback of the password field unsettled them. They are used to writing a few letters and then take a look at what they have written, which is no longer possible. Also, they would not notice when they had mistyped a letter. In case they did, they had problems deleting the right amount of letters, again due to the missing visual feedback.

While aware of the difficulties, it was not expected to pose such a challenge for the children. Otherwise, additional options like placing a checkbox to show the password in plaintext would have been taken into consideration. This is a common feature for handheld devices, however, due to security issues, it was not implemented. Children would be able to see each other’s passwords easily as they would be sitting close together. Still, it is considered as an option in future versions.

4.4 The Connection Process

The next two figures display the connection process. As illustrated earlier in figure 3 (Menu Screen), the player has two choices, either to start a new game, or to join an already existing one.

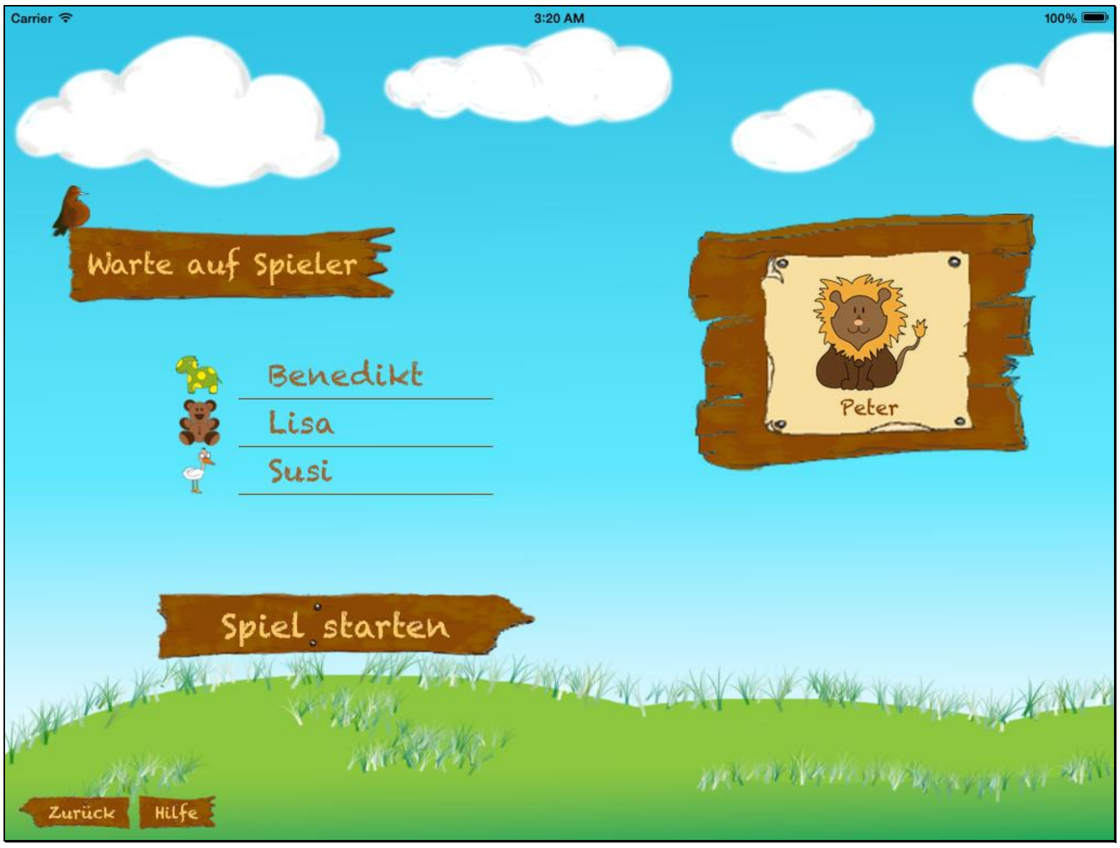


Figure 7 Start Game Screen

Both screens are designed in similar fashion. The Start Game Screen[8] (figure 7) appears when a player opens a new game and shows the list of people which have already joined the game. A total of three slots are available. The player who opened the game can decide to start it whenever he/she wishes. It is also possible to start a game without other players. In this case a confirmation dialog will appear, asking the player if he really wants to play alone.

[8] Translation of the screen elements, from top to bottom: label “Waiting for Players, button “Start Game”, button “Back”, button “Help”

The Join Game Screen[9] (figure 8) displays a list of open games.

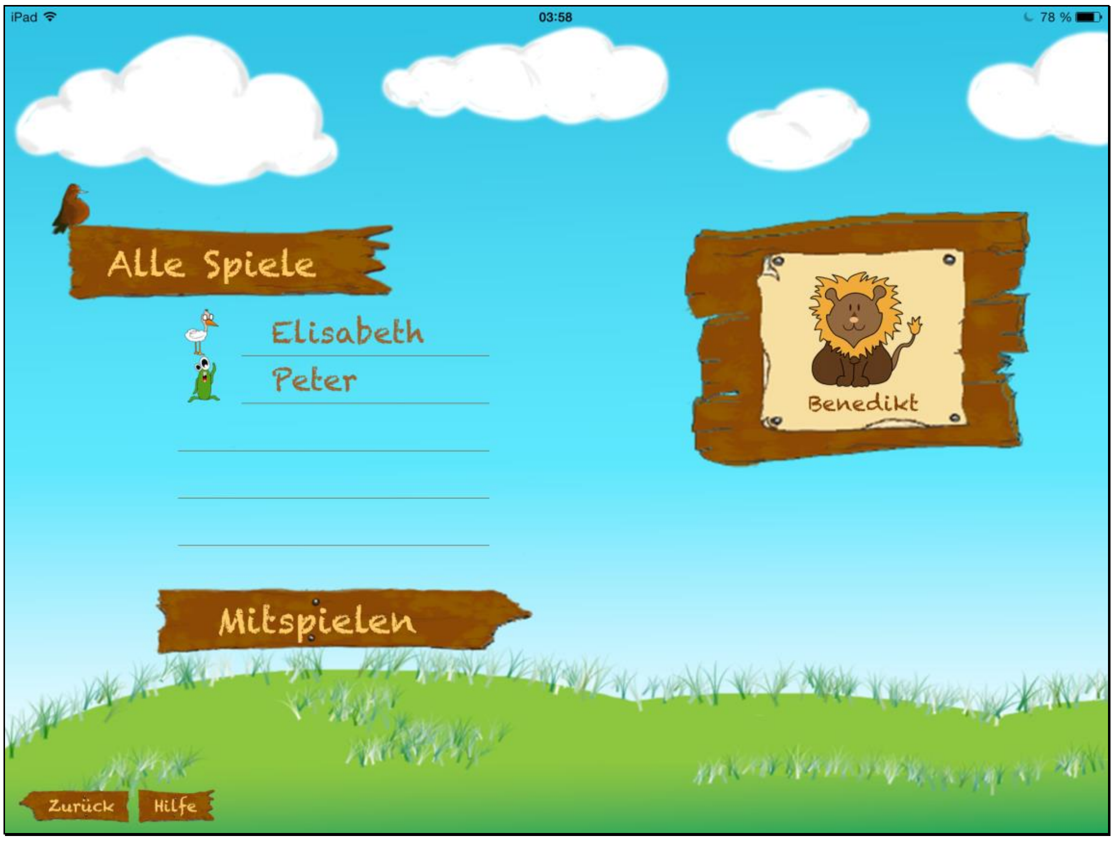


Figure 8 Join Game Screen

The player has to select one game and hit the “Join” button. A bubble will appear, informing the player that he/she has successfully connected and has to wait for additional players to join the game. The game will commence once the server decides that all players are present and starts the game.

[9] Translation of the screen elements, from top to bottom: label „All Games“, button „Join Game“, button „Back“, button „Help“

4.5 The Game Screen

This is the main screen[10] (figure 9) of the app.

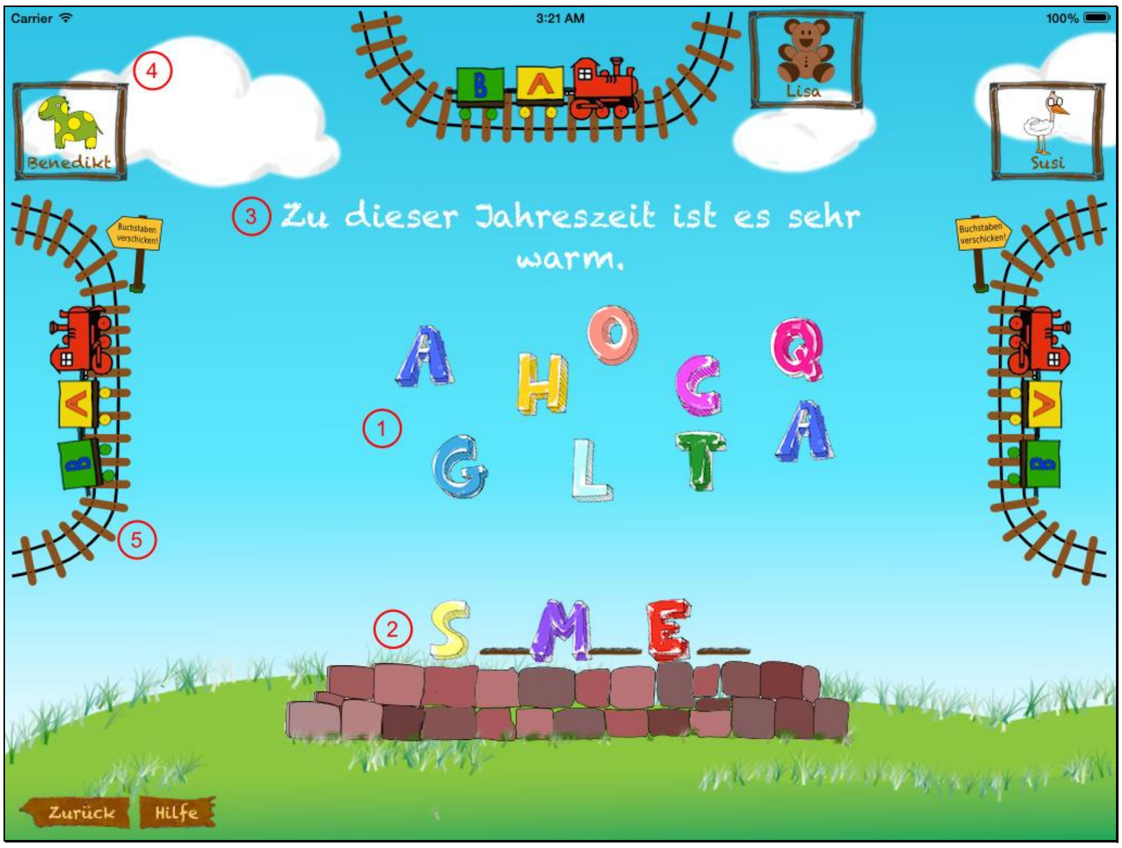


Figure 9 Game Screen

The middle of the screen contains the available letters (1), below, resting on the stone wall, is the word-to-guess (2). A hint for the word (3) is displayed in white letters above the middle section.

On the borders of the display, next to the railways, are the icons (4) of the teammates located. These icons can be switched (by drag and drop), so that they reflect the sitting position of the players on table.

At the bottom left side are two buttons, “Return” and “Help”, which are displayed in every screen except the first one. The “Return” button will end the game and take

[10] Translation of the screen elements, from top to bottom: textfield (white) „At this time of year it is very hot.“, yellow railway signs “Send Letter”, word-to-guess “Summer”, button “Back”, button “Help”

the player back to the start screen. The help button displays a help message in combination with arrows and images to further explain the gaming mechanism and functionality of the screen.

The first goal of the game is to fill in the missing letters of the incomplete word, by drag and dropping the letters into the right position. The second part is to help other players finish their words by sending them letters. To do that, the player has to drag the letter on one of the trains (5) or the area around. The icon (4) above the train represents the player the letter is sent to.

When all players have finished their words, a new round is initiated. The game server will semi-randomly draw words out of the downloaded word-pool (with priorities for new / unfinished words) and spread them among the players.

The game ends when there are no more words in the word-pool, or the player who launched the games closes it. If any of the other players leave, the missing letters will be spread among the reaming teammates and the game continues.

A few remarks on the design of the screen. Children prefer colourful designs with multimedia content (Large, et al., 2002) (Naidu, 2005) (Budiu & Nielson, 2010). The screen was designed in a way that would look appealing for children, yet not too overloaded with graphics, which would be confusing. Therefore, the main area of the screen is held in simple blue colours and free from other graphics, so that the children can focus on the question, letters and word. To make it more entertaining, simple animations were added whenever possible. For example, the train wagons will move along the rails, there is an animation whenver a letter is placed in the right place or not, another one rewarding the players when they finish a word, and so on. All animations are accompanied by sound.

This chapter should shed some light on the thoughts and decisions regarding the design of the application and explain the features and gaming mechanism. The next chapter will be a technical one, focussing on the details of the implementation. Each class will be inspected and essential pieces of code will be highlighted.

5 System Architecture and Implementation

5.1 Technical Framework

This section discusses why iOS was used as operating system. In addition, a short overview of the used programming languages, frameworks and libraries will be given. Theoretical background of the technologies that were deployed will be provided along the way.

5.1.1 iOS as a Platform

The app was developed for iOS and for iPads in particular. iOS was used as a platform for several reasons, the two vital ones being:

1. TU Graz has a long history developing educational apps[11] for iOS. The lecture “Mobile Applications” has been held since 2010 and during the last three years over 100 iOS applications have been developed by students, including a considerable number auf educational apps and games.
2. While the insights of those lectures could be applied for app developed regarding every platform, it seems that iPads were the preferred choice of schools during the last years and several so called “iPad classes” exist in Austria. While other operating systems are gaining in popularity among schools, considering that only a very limited number of schools provide tablets for school children at all, iOS still appeared to be the most reasonable choice.

[11] All apps are available at http://app.tugraz.at/

Hypothetically, it would have been possible to develop the app as cross-platform software. Using HTML5 as a programming language, the program would be portable to every device that has a current internet browser installed. However, in addition to several technical problems, it is questionable that such an implementation would make sense at all. For a more static application HTML 5 is a great way to achieve cross-platform availability. The app, being a game for children, has much more interactive content, as well as animations that would never run as smoothly in HTML 5 as they would in a native iOS (or Android, etc.) program. Furthermore, the concept of peer-to-peer connectivity would have to be realized in a less than optimal way. In a native app, it is possible to use either WLAN or Bluetooth to establish a direct peer-to-peer connection. Over HTML the communication between the devices would have to be relayed over a webserver, thus requiring an internet connection to be able to play.

5.1.2 Programming Language, Frameworks and Libraries

“Buchstaben Post” is a native iOS application. The programming language for iOS is called Objective C, which is an advanced, object oriented version of the original language “C”.

The app was developed for iPads only. As an alternative, a so called “Universal App” could have been built, meaning that it would run on both devices, iPhones and iPads. While this would have made the program available for a much wider range of users, it was not possible due to user-interface restrictions. The iPhones’ screen size is too small to display the required content.

To develop applications for iOS, Apple has provided a custom IDE[12] called Xcode. There is no other way to develop software for iOS. The IDE provides everything that is needed to compile, debug, and test an application. For debugging purposes, Xcode comes along with a simulator for iPhones and iPads. This means that it is possible to test an application on a desktop computer the same way as if it would be running on a real iPad or iPhone. However, for reasons unknown, Apple allows only one instance of the simulator to be executed at the same time. For most purposes, this would be enough, in case of a multiplayer game with 4 people it severely limits the testing capabilities, unless you are in possession of more than one real device. This has been a major problem throughout the whole development process of the app and is further discussed in a dedicated section of this chapter.

[12] IDE stands for Integrated (or Interactive) Development Environment

In addition to standard resources, several other frameworks provided by Xcode were used. They are listed below with a short explanation of each one:

- *GameKit*[13]. This framework is an essential part of Apples “Game Center”, which is a collection of interconnected components, composed of the following:
    - The Game Center Service is the online part of Game Center, where it is possible to store player and game data on a destinated server.
    - The Game Kit Framework, which provides the classes necessary for developing games.
    - The Game Center application provides a centralized app that players use to access Game Center’s features.

  As it can be inferred from the description above, Game Center is essentially an online service and thus not an option for this application, which should be able to function without internet connection. However, a part of the GameKit framework can be used to establish peer-to-peer connections over WLAN and Bluetooth. The GKSession class allows the direct connection of devices in the same network or in Bluetooth proximity and provides a peer-to-peer connection mode as well as a server-client infrastructure.
- *QuartzCore*. Also known as Core Animation. It is an Objective-C framework that supports image processing and video image manipulation.
- *AVFoundation*. The framework provides an interface for managing and playing audio-visual media.
- *CoreGraphics* is an API based on the Quartz drawing engine, providing low-level support for 2D rendering. It can handle tasks like path-based drawing, transformations and colour management.

As mentioned before, the game can be played offline or online. In the latter case, the user has to sign in, which requires an account at the user management system. The website, user manager and game database will be explained in full length in the following chapter, for the moment it should be enough to note that there exists a database where the usernames, school classes and custom wordlists are stored.

[13] https://developer.apple.com/library/ios/documentation/NetworkingInternet/Conceptual/GameKitGuide/Introduction/Introduction.html (Last visited: Nov. 2013)

No direct connection to the database is established from within the app, all communication is relayed over the web-interface. For this purpose, Apples URL[14] Loading System is employed. The library provides a simple interface to dispatch URL requests. It was decided to encode the exchanged data in XML[15].

Establishing a peer to peer connection was another challenge, from the technical perspective as well as from a design point of view. The game was intended for school classes where each pupil is in possession of his own tablet device. While the children might have acquired a general proficiency in using the operating system, they will not have a concept of how a WLAN or Bluetooth connection works, or, for that matter, how the devices are connected at all.

To address the problem, the application is able to connect over WLAN as well as Bluetooth. If one method is turned off or not able to establish a connection, the application will automatically try to connect over the other medium. Luckily, the Gamekit framework supports that kind of connection management. Only the interface had to be adapted for the use of children. While that solved problems concerning the functionality of the connection process, several technical issues regarding the connection process of the GameKit framework had to be overcome. Through a tedious process of trial and error during which several frameworks for Bluetooth and WLAN connection management were tested, it was confirmed that the GameKit framework was best suited for the purpose of this application. Nevertheless, during the first field test with more than 10 devices, new problems arose which could not be located during the development phase of the application due to a lack of physical test devices.

[14] Uniform Resource Locator (URL)

[15] Extensible Markup Language (XML)

5.2 System Design and Architecture

Figure 10 illustrates the architecture of the system.

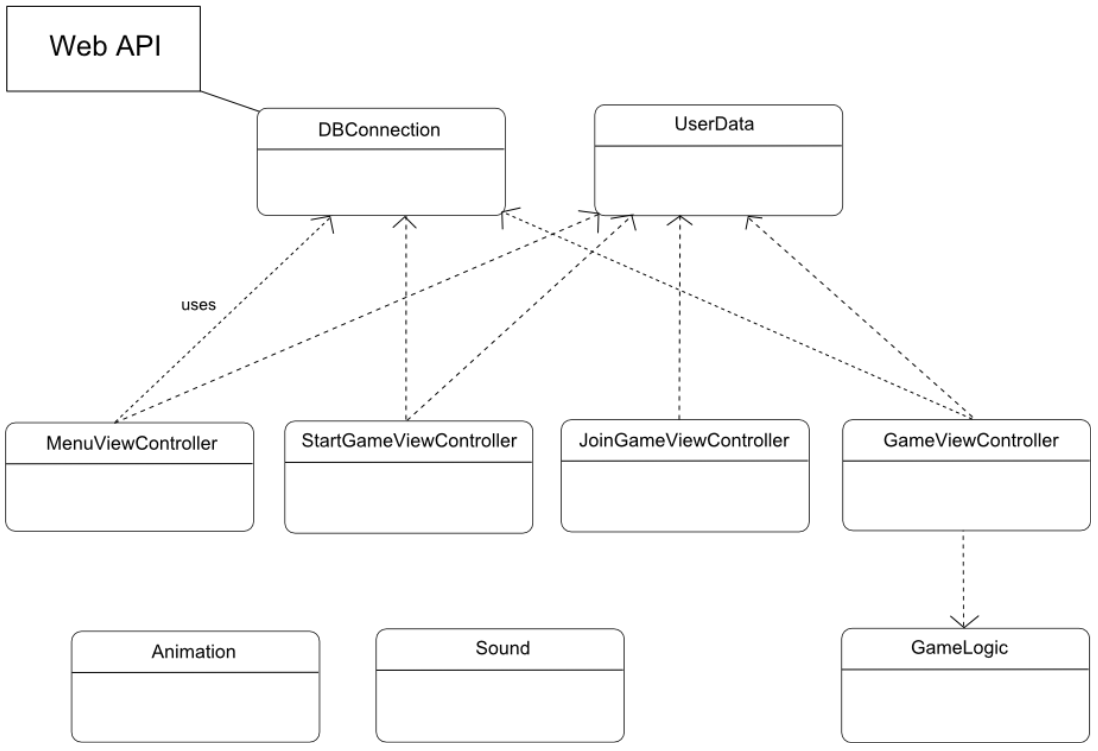


Figure 10 System Architecture

The app is composed of four screens (not counting the splash or loading screen), those screens are represented through their corresponding ViewController classes. In each ViewController, the functionality of this part (or screen) of the application is implemented. DBConnection is an interface which provides methods to communicate with the web-interface. UserData is a Singleton that stores individual data, like username and icon. GameLogic implements functions regarding the game mechanism, like word generation. The Animation class holds all the code regarding image manipulation. Sound provides methods to play mp3 files.

5.3 Xcode and Objective C

This section provides information on Xcode, Objective C and the build settings of the application.

5.3.1 Xcode

Xcode[16] is an integrated development environment (IDE) provided by Apple. It is available free of charge in the Mac App Store for OS X users. Xcode contains a suite of development tools for the creation of Mac, iPhone, iPad and iPod touch applications. The supported programming languages are C, C++ and Objective C.

Xcode has a number of useful tools. For the purpose of this project the following tools were used:

- *XCode IDE* for the management, compiling and debugging of source code.
- *Interface Builder* for the creation of graphical interfaces.
- *Instruments*, which is a tool to analyse memory usage, data access and program performance.
- *iPhone/iPad Simulator* for testing purposes.

5.3.2 Objective C

Objective C is an object oriented expansion of the original programming language C. Every program that is written in C can be compiled with an Objective C compiler. Among the most prominent features of Objective C are the following (Lee, 2013):

- *Object Orientation.* Complete support of object orientated principles, including object messaging, encapsulation, inheritance, polymorphism and open recursion.

[16] https://developer.apple.com/technologies/tools/ (Last visited: Nov. 2013)

- *Object messaging.* Objects can pass messages, the receiver uses the message to invoke the corresponding method if it exists or handles it in a number of different ways.
- *Memory management.* In newer versions, Automated Reference Counting (ARC) is supported, which is an automatic memory management mechanism (garbage collection, etc.).
- *C language support.* Objective C is a superset of C, meaning that all C libraries can be directly accessed.
- *Software libraries.* Apple provides numerous software libraries which ease the development of applications significantly.

Objective C is the primary language of the Cocoa and Cocoa Touch framework, which is comprised of the key frameworks needed to develop applications for iOS or Mac OS:

- *Foundation Kit Framework*
- *UI Kit Framework*
- *Game Kit Framework*
- *iAd Framework*
- *Map Kit Framework*

Among those, the Foundation Kit, UI Kit and Game Kit framework are used in the app. Some of the main features of Cocoa Touch are Gesture Recognizers, Multitasking and Animation[17].

5.3.3 Build Settings

The app was developed and deployed for iOS 6.0 or higher, meaning that devices which are running version 4.x or 5.x won't be able to install the app at all. This excludes all users of the first generation iPad from installing the program, since it is not possible to upgrade those devices above iOS version 5.1.1. However, due to apples strict upgrade policy, only a very small amount of devices remains below version 6.x:

[17] http://en.wikipedia.org/wiki/Cocoa_Touch (Last visited: Nov. 2013)

Table 1 iOS Version Distribution[18] (Nov 22, 2013)

iOS Version	**In Use**
7.X	67.4 %
6.X	26.0 %
5.X	6.1 %
4.X	0.6 %

iOS is a constantly changing and evolving and with every major change of the operating system, programming libraries and Xcode features change as well. While this constantly adds features and improves the development environment, it also means that programs have to be adapted continuously and that many features will not be available in lower versions of the operating system. Due to those reasons and considering the statistic above - 93 % of all users have iOS 6.X or higher installed – it was decided to deploy the app for iOS 6.0 and above.

The app was developed for iPads only. It will run on all iPads above version 1 as well as on the iPad mini. All graphics were designed for the use of Retina-Displays. Starting with the third generation of iPads, all devices are equipped with Apple’s so called Retina-Display, except the iPad Mini. Displays carrying that label are built with a very high pixel density, for iPads this adds up to 264 pixels per inch at a resolution of 2048 x 1536 on a 9.7 inch display. Thus, fore every graphic, two versions have to be designed, one for the use with normal screens at a resolution of 1024 x 768 pixels as well as a separate version for Retina-Displays. In addition, the available screen size within the app varies from version 6.X to 7.X, meaning that some of the graphics have to be available in two more versions, now already adding up to four different image sizes. This should point out one of the difficulties in developing apps for mobile devices. Regarding iOS, this is a minor problem, because of the limited number of devices which are all produced by the same company. However, it is an altogether different matter when developing for Android or other freely available operating systems, where manufactures supply us with an almost indefinite number of screen sizes and resolutions. Usually the problem can be avoided through the use of relative layouts, but this is not always possible in graphic oriented applications. This should by no means be misinterpreted as a critic against other mobile operating systems. Nev-

[18] http://david-smith.org/iosversionstats (Last visited: Nov. 2013)

ertheless it is a fact that has to be considered when developing graphic-oriented applications for mobile devices.

5.4 Detailed Description of the Implementation

Starting with an overview of the significant methods of every class, theoretical background on employed technologies and design patterns will be provided. After that, important pieces of the implementation will be explained in detail.

5.4.1 Connection to the Web-Interface (Class DBConnection)

DBConnection represents the interface to the website. It provides methods to authenticate the user, download the wordpool and upload which words a player has finished.

5.4.1.1 Class Overview

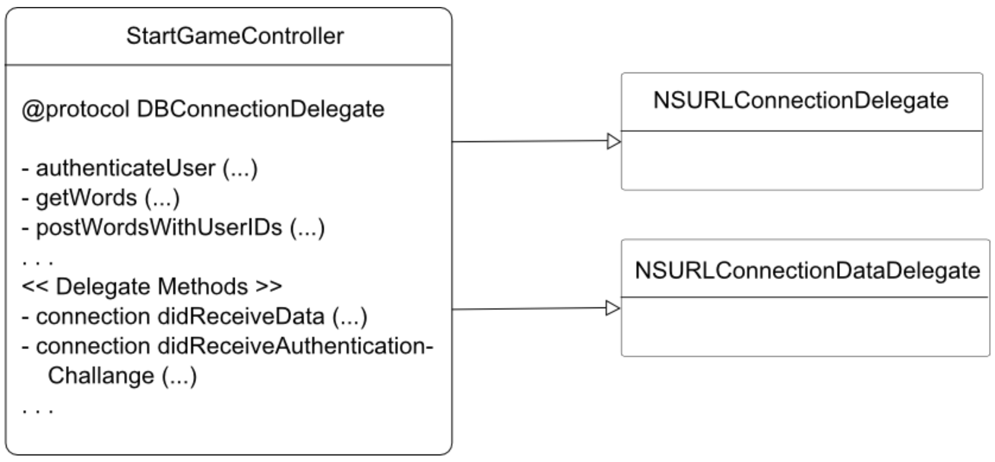

StartGameController
@protocol DBConnectionDelegate
- authenticateUser (...)
- getWords (...)
- postWordsWithUserIDs (...)
. . .
<< Delegate Methods >>
- connection didReceiveData (...)
- connection didReceiveAuthentication-Challange (...)
. . .
NSURLConnectionDelegate
NSURLConnectionDataDelegate

Figure 11 DBConnection

The class implements the Delegate Protocol, a protocol or pattern that is very common in the Objective C and the Cocoa framework. It is almost impossible to develop applications for iOS without an understanding of the protocol, therefore it will be explained in detail.

5.4.1.2 The Delegate Protocol

Delegation describes a powerful concept where one object assigns tasks to another object, its “delegate”. This is analogous to subclasses deferring requests to parent classes, with the difference that the receiver passes itself to the delegate to let the delegated operation refer to the receiver (Gamma, et al., 1994).

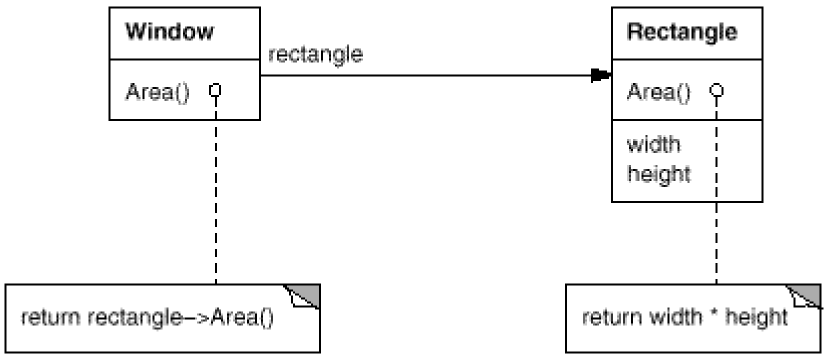


Figure 12 Delegation (Gamma, et al., 1994)

This is an example for delegation, the Window class is in possession of an instance variable of Rectangle and delegating Rectangle-specific behaviour to it.

The main advantage of delegation over inheritance or composition is that it is easy to change the delegate at runtime. Rectangle could be easily replaced by a class called Circle, as long as it implements the same method Area (). This would not be possible if Window was a subclass of Rectangle (Gamma, et al., 1994).

Returning to the implementation, DBConnection both implements the delegate protocol itself as well as the interfaces for the NSURLConnection delegates. This is just a means of passing along tasks. In the case of this app one possible scenario would be the following:

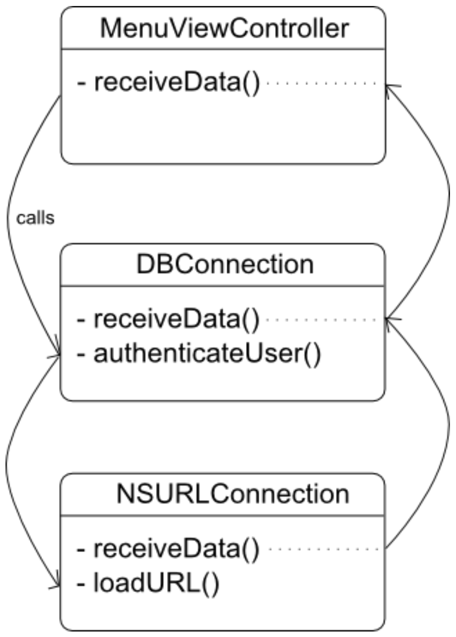


Figure 13 Delegate DBConnection

The diagram illustrates what happens when a user wants to sign in. The class MenuViewController calls the function authenticateUser() of DBConnection which in turn will call loadURL() from NSURLConnection. When the NSURLConnection object has finished loading, the delegate method receiveData() from DBConnection will be called and so on.

5.4.1.3 NSURLRequest

The NSURLRequest class is used to post and receive data from the web-interface. The object represents an URL load request independent of protocol and URL scheme. It encapsulates two basic elements of a load request: the URL to load and the policy to use[19].

5.4.1.4 HTTP Basic Access Authentication

HTTP-Authentication is a simple method to enforce access control of web resources. It uses standard HTTP headers, no handshakes have to be done and it doesn’t require cookies, session identifier or login pages. The transmitted credentials

[19] https://developer.apple.com/library/mac/documentation/Cocoa/Reference/Foundation/Classes/NSURLRequest_Class/Reference/Reference.html (Last visited: Nov. 2013)

are BASE64 encoded, but not encrypted or hashed in any way. Basic Access Authentication provides no confidentiality protection for the transmitted credentials[20].

On the server side, the request will be sent in form of a HTTP 401 Not Authorized response code containing a WWW-Authenticate HTTP header.

The client will send the server its authentication credentials through the authorization header, which is constructed of the username and password, encoded in Base64 and the authentication method (basic)[21].

5.4.1.5 Implemetation Details

The methods listed in the class diagram will now be explained in more detail.

Listing 1 User Authentication

```
- (BOOL) authenticateUser: (NSString*) strName Password: (NSString*) strPass-
word {

       NSString* pwdHash = [self sha256:strPassword];

       NSURL *url = [NSURL URLWithString: [NSString stringWithFor-
       mat:@"%@%@", strServerURL, @"authenticateUser"]];


      NSMutableURLRequest __autoreleasing *request = [[NSMutableURLRequest
      alloc] initWithURL: url];

      [request setHTTPMethod:@"POST"];

      NSString *postString = [NSString stringWithFor-
      mat:@"username=%@&password=%@", strName, pwdHash];

      . . .

      NSURLConnection __autoreleasing *connection = [[NSURLConnection alloc]
      initWithRequest:request delegate:self];

      . . . }
```

The important pieces of code are marked in green. The method receives username and password and creates an NSURLRequest. To achieve a minimum of confidentiality, the password is hashed with the SHA256 algorithm. Next, the request HTTP method is set to post and provide the username and password hash as string. The request is then passed to the URLConnection object which will initialize a connection. Here the delegate pattern can be seen at work. DBConnection implements the delegate methods of URLConnection and passes its reference along to the connection object. Now, when the connection starts to load the URL and receives data,

[20] http://en.wikipedia.org/wiki/Basic_access_authentication (Last visited: Nov. 2013)
[21] http://tools.ietf.org/html/rfc1945#section-10.16 (Last visited: Nov. 2013)

the according delegate method in DBConnection is called and the data is passed along as a parameter.

This will post the user credentials to the web-interface, which will try to authenticate the user at the user management system of Graz University of Technology[22]. A successful authentication will return the ID of the user.

However, in order to post data to the web-interface, the application has to authenticate itself via Basic HTTP authentication. A delegate method of NSURLConnection is called whenever the connection object receives an authentication challenge.

Listing 2 Authentication Request

```
- (void)connection:(NSURLConnection *)connection didReceiveAuthentication-
Challenge:(NSURLAuthenticationChallenge *)challenge {

    if ([challenge previousFailureCount] == 0) {

       NSURLCredential *newCredential = [NSURLCredential

           credentialWithUser:app_name

               password:app_pwd

                 persistence:NSURLCredentialPersistenceForSession];

      [[challenge sender] useCredential:newCredential

           forAuthenticationChallenge:challenge];

    }

}
```

While not the most secure authentication - the login credentials are not significantly protected - it is very easy to implement on the webserver as well as in Objective C. This snipped of code is all it takes on the client side, the part on the server side is even smaller.

This should explain the communication between app and webserver, all other methods in this class work the same way. For example, to download the wordpool, a URL request posts the ID of the user and in return a XML file which contains the wordlist assigned to the user is downloaded.

[22] The user management system is currently located at mathe.tugraz.at (Nov. 2013).

5.4.2 User Specific Data (Class UserData)

As the name implies, the class will be used to store user specific data like username, id, icon, and so on.

5.4.2.1 Class Overview

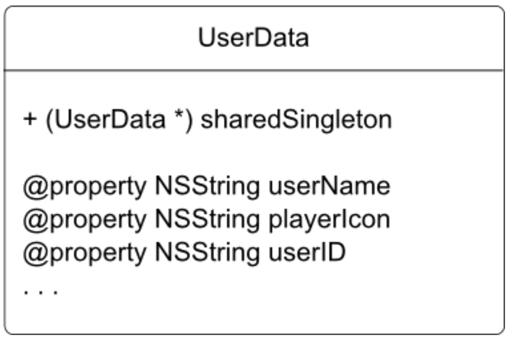

UserData
+ (UserData *) sharedSingleton
@property NSString userName
@property NSString playerIcon
@property NSString userID
. . .

Figure 14 Classdiagram UserData

The variables username, icon and userID will be required in every view of the program, thus it would make sense to make them globally available. However, as global variables are usually something to avoid, it was decided to create a class called UserData that implements the Singleton pattern, which has some benefits over global variables.

5.4.2.2 Singleton Pattern

The intent of the pattern is to ensure that a class has only one instance, and to provide a global point of access to it (Gamma, et al., 1994).

The structure of the pattern is illustrated in the following diagram.

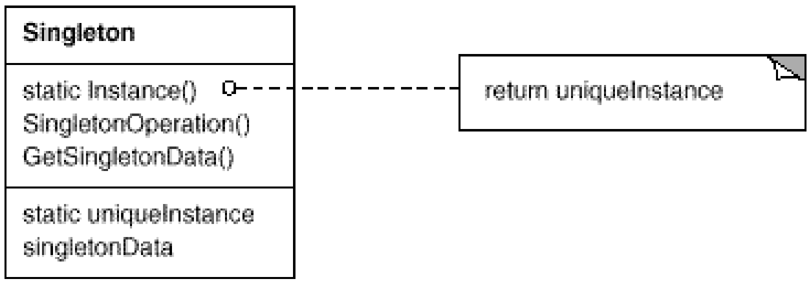


Figure 15 Singleton Pattern (Gamma, et al., 1994)

The pattern is a simple one. It defines an operation that lets the client access its unique instance. The advantages are the following (Gamma, et al., 1994):

- Controlled access to sole instance.
- Reduced name space. Avoids polluting the namespace with global variables.
- Permits a variable number of instances (does not have to be only one).
- More flexible than class operations (i.e. static member functions).

5.4.2.3 Implementation Details

The only interesting part of code in this class is the method which returns the Singleton instance.

Listing 3 Singleton

```
+ (UserData *)sharedSingleton {
    @synchronized(shared)
    {
        if ( !shared || shared == NULL )
        {
            shared = [[UserData alloc] init];
        }
        return shared;
    }
}
```

The “synchronised” keyword declares a critical section around the code block. In multithreaded code “synchronised” guarantees that only one thread can be executing this code block at any given time.

To access the object at some point in the program, all that has to be done is calling the following method and it will return the instance.

Listing 4 Access Singleton

```
UserData* usrData = [UserData sharedSingleton];
```

More functionality was added to the UserData object besides providing access to user variables. When the object is initialized, previously stored settings (username and icon) are loaded from a local file.

An easy way to store data on the device in an organized way is to use information property lists[23] (plist). A plist file is a text file that is typically UTF-8 encoded and the contents are structured using XML. Objective C provides easy read and write access.

To sum it up, the class implements the Singleton pattern, meaning that only one instance can be created and there is a global point of access. It is used to store user data and to read and write this data to a plist file located on the device.

5.4.3 Animation (Class Animations)

This class deals with all animated content.

5.4.3.1 Class Overview

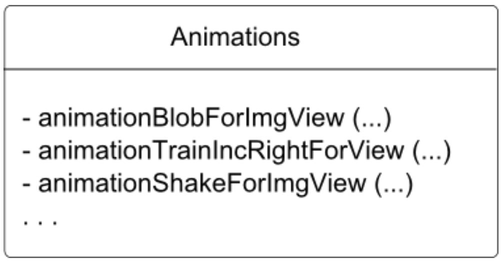

Animations
- animationBlobForImgView (...)
- animationTrainIncRightForView (...)
- animationShakeForImgView (...)
. . .

Figure 16 Classdiagram Animations

Two different libraries for animations (UIKit and QuartzCore) are used, depending on the kind of animation that has to be performed. Animations are probably a good way to point out one of the greatest benefits of Xcode or Objective C and the Coca Touch framework. As it is the case with every IDE and programming language, each one has its unique advantages and disadvantages. Objective C has a rather uncommon syntax that takes some time getting used to, for example. In any case, one of

[23] https://developer.apple.com/library/ios/documentation/general/Reference/InfoPlistKey Reference/Articles/AboutInformationPropertyListFiles.html (Last visited: Nov. 2013)

the big advantages certainly comes in form of the numerous libraries and frameworks that Objective C and the Cocoa have to offer.

An animation for an image or for any subclass of UIView is set up in just a few steps:

- Create the animation by setting parameters like duration and delay.
- Additional parameters like animation speed (linear, curve, etc.) can be set.
- Specify the animation property (more than one value possible). Properties of UIView that can be animated are size and position, transparency and transformations (scaling, rotation, etc.).
- Animations can be executed in sequence.

The following piece of code moves an UIImageView from its current position to a specific point.

Listing 5 Animations

```
- (void) animationMoveImg: (UIImageView *) imgView toLocation: (CGPoint) lo-
cation inTime: (int) time
{
      [UIView animateWithDuration:time
                              delay: 1.0
                              options: UIViewAnimationCurveEaseInOut
                              animations:^{  imgView.center = location;  }
                              completion:^(BOOL finished){  }];
}
```

This is all it takes to create an animation as explained above. The last option parameter “completion” makes it possible execute another animation after the current one has finished. This way, a sequence of animations can be performed.

While it is possible to move an object from one position to another, the moving path itself cannot be specified.

The CALayer class from the QuartzCore framework offers another way to animate content. The class itself manages image-based content and performs animations on that content[24]. It is used to create an animation that allows an image to move along a specified path.

Listing 6 Advanced Animations

```
- (void) animationTrainIncLeftForView: (UIView*) animView {
    UIBezierPath *trackPath = [UIBezierPath bezierPath];
    [trackPath moveToPoint:P(0, 500)];
    [trackPath addQuadCurveToPoint:P(70, 420) controlPoint:P(70, 500)];
    [trackPath addLineToPoint:P(70, 310)];

    CALayer *lockomotive = [CALayer layer];
    . . .
   [self animationAlongPath:animView afterDelay:1 withLayer:lockomotive for-
   Path:trackPath inTime:3];
   . . .
}
```

A path is defined by a starting point and multiple path segments. A segment can be specified in form of a Bezier curve or as a straight line.

5.4.4 Sound (Class Sounds)

One way to add sounds to an app is the use of the AVFoundation framework. Again, only a few lines of code are needed.

Listing 7 Sound

```
- (void) playSound:(NSString *)filename
{
    NSString *path = [[NSBundle mainBundle] pathForResource:filename
   ofType:@"mp3"];
   audioPlayer=[[AVAudioPlayer alloc] initWithContentsOfURL:[NSURL fileURL-
   WithPath:path] error:NULL];
   [audioPlayer play];
}
```

[24] https://developer.apple.com/library/mac/documentation/GraphicsImaging/Reference/CALayer_class/Introduction/Introduction.html (Last visited: Nov. 2013)

This snipped of code makes it possible to play files of the type mp3. Other formats are supported as well. The reason why this piece of code was included is to further illustrate the point from before. A lot of common tasks, especially regarding multimedia, gesture recognition and similar activities that would otherwise be very difficult or time consuming to implement can be created with only a few lines of code.

5.4.5 The Menu Screen (Class MenuViewController)

This class contains all the functionality regarding the menu (or start) screen of the app.

5.4.5.1 Class Overview

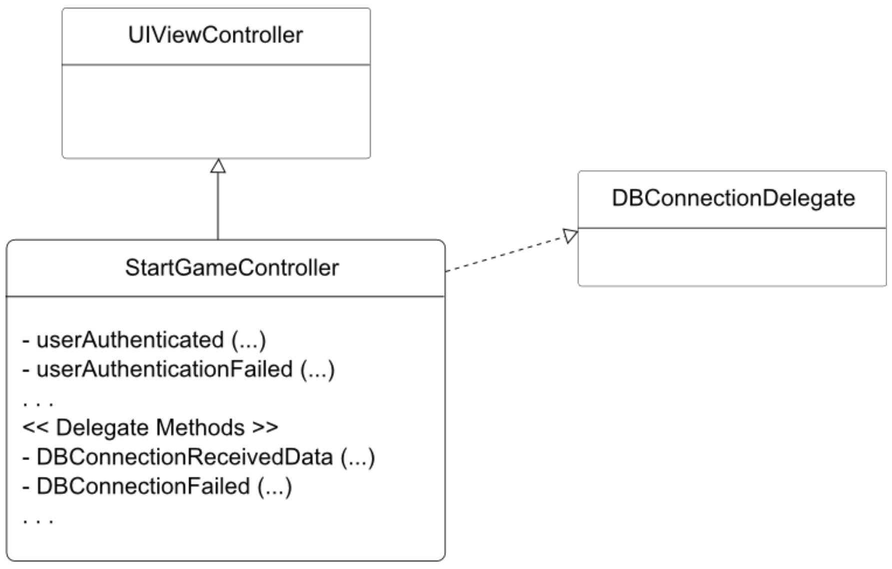

UIViewController
DBConnectionDelegate
StartGameController
- userAuthenticated (...)
- userAuthenticationFailed (...)
. . .
<< Delegate Methods >>
- DBConnectionReceivedData (...)
- DBConnectionFailed (...)
. . .

Figure 17 Class Diagram MenuViewController

As it can be seen in the class diagram, MenuViewController is a subclass of UIViewcontroller which manages all views of the associated screen. The UIViewController is the most essential part of an application, every app contains at least one custom subclass of it.

5.4.5.2 UIViewController

The UIViewController class provides the fundamental view-management model for all iOS apps[25]. It is responsible for:

- Layout and resizing of its views.
- Adjusting the contents of the views.
- Acts on behalf of the views when the user interacts with them.

The class is tightly bound to its views and takes part in handling events.

5.4.5.3 Storyboard

Storyboard is an essential feature of Xcode that provides a visual representation of the user interface of an app. It shows screens and its contents and the connection between the screens. A scene (or screen) represents a view controller and its views. Scenes are connected by segue objects, which represent a transition between two view controllers[26]. Segues should be used for two things:

- To set the type of transition between to scenes (e.g. model or push).
- To pass along data between the scenes and the corresponding view controllers.

The storyboard feature was introduced with iOS 5.0.

5.4.5.4 UINavigationController

The navigation controller manages the screens (view controllers) and the hierarchy in which they are displayed. View controllers are placed on a navigation stack. The root view controller, which is the starting point of an app, is at the bottom, the

[25] https://developer.apple.com/LIBRARY/IOS/documentation/UIKit/Reference/ UIViewController_Class/Reference/Reference.html (Last visited: Nov. 2013)

[26] https://developer.apple.com/library/ios/documentation/general/conceptual/Devpedia-CocoaApp/Storyboard.html (Last visited: Nov. 2013)

currently displayed view controller at the top. When navigating back and forth between screens, view controllers are constantly pushed and popped from the stack[27].

5.4.5.5 Event Handling

Events are objects sent to an app to inform it of user actions. Events can take on many forms: Single-Tap, Multi-Touch and motion events, as well as events for controlling multimedia. Common gestures are built into the UIKit framework and send an action message to the target object when the event occurs. A more powerful way to handle events are gesture recognizers. A gesture recognizer can be attached to a view, and the entire view will act like a control element, responding to whatever gesture specified[28].

An example of each method will be given in the following section.

5.4.5.6 Implementation Details

There are no specifically interesting pieces of code in this class, except from event handling. The user authentication procedure was already explained in the previous section (5.4.1). The menu view controller only provides the graphical interface for text input in form a login view.

Listing 8 Events

```
- (IBAction) btnPressEvent: (id)sender {   …   }
```

This is the fastest way to handle events. All that has to be done is to link an element (button, image, etc.) with the corresponding action method. This can be accomplished in the interface builder of Xcode.

In the next example, the menu view controller is added as delegate to a text field. Whenever an event is triggered, the corresponding delegate methods will be called.

[27] https://developer.apple.com/Library/ios/documentation/UIKit/Reference/ UINavigationController_Class/Reference/Reference.html (Last visited: Nov. 2013)

[28] https://developer.apple.com/library/ios/documentation/eventhandling/conceptual/ eventhandlingiphoneos/eventhandlingiphoneos.pdf (Last visited: Nov. 2013)

Listing 9 Events as Delegates

```
    self.txtName.delegate = self;
    …
- (void) keyboardDidShow: (NSNotification *) notification { … }
```

5.4.6 Establishing a Connection (Class StartGameViewController)

The main responsibilities of this class are to establish a connection with other players and to download the wordpool for the game, if the player is logged in.

5.4.6.1 Class Overview

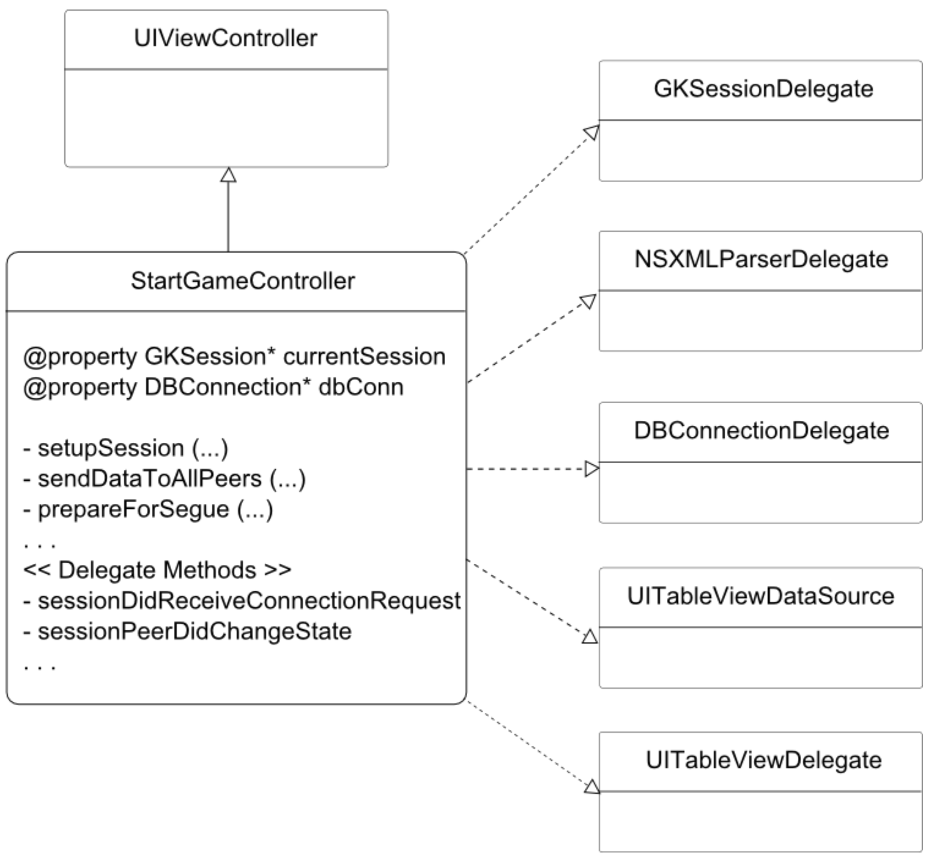


Figure 18 Class Diagram StartGameViewController

The interesting part of this class is the GKSession object, which will be explained in detail. Some of the problems with the GKSession class will be outlined and also which libraries could have been used as an alternative.

Before a game is initiated, the wordpool has to be downloaded via DBConnection. It is structured in XML, thus the need to implement the NSXMLParserDelegate to parse the XML content.

5.4.6.2 GameKit's GKSession Class

The GKSession class is part of the GameKit framework and provides the ability to discover and connect to nearby devices using Bluetooth or Wifi. A device is identified by its peer ID. Peers discover other peers by using a unique string to identify the service they implement, called a session ID.

Sessions can operate in two modes. A peer can either be configured as a server or client, or act as both server and client. A peer configured as server broadcasts its session ID, the client searches for other peers advertising the same session ID.

The GKSessionDelegate protocol has to be implemented to control the behaviour of the GKSession object. The delegate is called when remote peers are discovered, a connection is initiated or the state of any discovered or connected peer changes.

In addition, a data handler has to be provided so that the session can forward data it receives from connected peers[29].

The GameKit framework provides us with a standard user interface (GKPeerPickerController) for the discovery and connection procedure of devices using the GKSession class. However, the interface is not suited for children and could not be used for the purpose of this application.

The GKSession class is deprecated since iOS 7.0. It has been replaced by the Multipeer Connectivity Framework. iOS 7.0 had not been released during the time the application was developed, therefore the GKSession class is still in use. An update to the Multipeer Connectivity Framework will be considered in future versions of the app due to some of the problems with the GKSession object.

5.4.6.3 Problems with GKSession and Other Frameworks

Establishing and maintaining a connection is certainly one of the most important parts of the application. Two factors were most important:

- A stable connection.

[29] https://developer.apple.com/library/ios/documentation/GameKit/Reference/GKSession_Class/Reference/Reference.html (Last visited: Nov. 2013)

- The discovery process of remote peers had to be reliable and fast. It would be confusing for the children if a game would not appear after more than a few seconds. The field study showed that children would rather start the game again if it did not appear within less than five to ten seconds.

Previous experiences (Ebner, et al., 2013) had revealed certain problems regarding the GKSession library.

- The GKPeerPickerController is not completely reliable.
- Discovery of devices in the same WLAN does not always work or takes too long (more than a minute).

The first problem was avoided by devising a custom connection interface. The second problem is not to be found in the implementation of the GKSession class itself. It seems to be a network problem due to restrictions or size. Restrictions would block broadcasting of the session ID. In in case of a bigger infrastructure with multiple access points and a lot of users, discovering remote peers takes too long.

The GKSession class is built upon Bonjour, a protocol that allows devices or applications to find each other on the network. It provides a way for an application to tell others what IP address and port they can connect to in order to communicate[30].

To ensure that the connection problems were not caused by some fault in GKSession's implementation, an app was built for the sole purpose of testing the Wi-Fi connection capabilities by directly implementing Bonjour. This did not solve the issue and affirmed the assumption that the problem was not caused by the GKSession class, but the network itself.

The other option was to connect the devices over Bluetooth as it is supported as well. However, there is no way to enable Bluetooth directly or indirectly from within the application and children would have a hard time activating it on their own. In earlier versions of iOS it was possible to either change settings directly from within the app or to provide a link to the settings page. Due to a more restrictive policy and security issues, recent versions of iOS do not allow this. While those restrictions can be bypassed, the app most likely would not pass Apple's review process before entering the store.

[30] http://mobileorchard.com/tutorial-networking-and-bonjour-on-iphone/ (Last visited: Nov. 2013)

During the two field tests at school classes with 15 devices and more, another problem was encountered. While connections over WLAN worked without problem in both cases, enabling Bluetooth on all devices caused connection failures. The exact cause could not be determined, it appeared to be a result of Bluetooth connections cancelling each other out. This phenomenon did not appear with a limited number of devices.

To sum up the benefits and disadvantages of Game Kit's GKSession Class:

- Provides an easy interface for developers to implement peer-to-peer connectivity.
- It performs well under normal circumstances.
- Supports Bluetooth and Wi-Fi.
- When using Wi-Fi, the size of the network can be a problem.
- Bluetooth seems to work only with a limited number of devices. Otherwise it might cause connection failures.

There is another workaround if neither Bluetooth nor Wi-Fi will work, however, it requires a bit of technical know-how and preparation from the teacher. An iPhone or iPad can act as a mobile hotspot, creating a WLAN access point. This can easily be configured in the settings page. The other devices just have to connect to the newly established network.

5.4.6.4 Extensible Markup Language (XML)

XML is a markup language much like HTML, however, it is not used to display data, but to store it. Data is presented in a hierarchical structure in the form of a text file. A XML file contains text symbols and should be easy to read for humans.

The structure of the wordlist will serve as an example:

Listing 10 Wordlist XML Structure

```
<wordlist>
   <item>
      <id>1</id>
      <question>This is a question?</question>
      <word>answer</word>
      <priority>1</priority>
   </item>
</wordlist>
```

A XML document is composed of elements (<wordlist>, <item>, etc.) which are marked by a start and end tag. Elements can have attributes in form of key-value pairs (e.g. <item id="1">). The document will have exactly one element at the top (root element), beneath it further elements can be nested (W3C, 2008).

5.4.6.5 Implementation Details

This section describes step-by-step what happens from the moment the class is loaded.

- Whenever a new view controller is pushed on the navigation stack, the method "viewDidLoad" is called. This method should be used to initialize variables and elements before the view is displayed. The session object is set up here.
- The following parameters will be specified:
  - Session ID. Every device will have the same session ID, so that they are able to discover one another.
  - A display name. This is an optional parameter which helps users to identify the devices. In this case, the display name contains the user's login name and a code for the selected user icon.
  - The session mode is set to GKSessionModeServer.
  - StartGameViewController is set as a delegate of GKSession.
  - The session is set to be available.
  - A connection timeout is defined.

- The session object is now advertising itself to any other device using the same session ID. As shown earlier in figure 8 (JoinGameViewController), players who are looking for open games will be able to see the game listed in the UITableView in form of the player’s icon and username.
- When a player decides to join the game, the StartGameViewController’s delegate method “session didReceiveConnectionRequest” will be called with the remote peer’s ID as a parameter. If the connection is accepted, both peers will be notified that their connection state changed to connected.
- Once all peers have connected and the player decides to start the game, a connection to the web-interface is established and the current wordpool is downloaded and parsed by the NSXML parser. The session objects availability is set to false so that the device will no longer be visible to other peers and the game is initiated.

There are several ways to pass data from one view controller to another, the following method is the recommended one:

Listing 11 Segues

```
-(void)prepareForSegue: (UIStoryboardSegue *) segue sender: (id)sender {
    if([segue.identifier isEqualToString:@"toGameView"]){
      GameViewController *gameview = (GameViewController*) se-
      gue.destinationViewController;
      gameview.currentSession = self.currentSession;
      gameview.isServer = TRUE;  }  }
```

A segue represents the connection between two view controllers. When switching from one view to another, the segue is performed and with it the method “prepareForSegue” is called. The function will hand the parameters to the new view controller before it is initiated (“viewDidLoad” method). In the marked piece of code, the session object is passed along and then the game view is told to act as a server.

The JoinGameViewController will not be explained, because JoinGame- and StartGameViewController are similar in most aspects. The only significant difference is that the session object will be running in client mode and there is no need to download a wordpool as it is done by the server. The client connects to the server and waits until the server sends a “start game” message.

5.5 The Game Screen (GameViewController)

The GameViewController can act as a client or server. In the latter case it will handle communication with the web-interface and see to the distribution of words, among other functions regarding the game mechanism. All functions concerning game logic that are not directly implemented in GameViewController are accumulated in the corresponding class GameLogic.

5.5.1.1 Class Overview

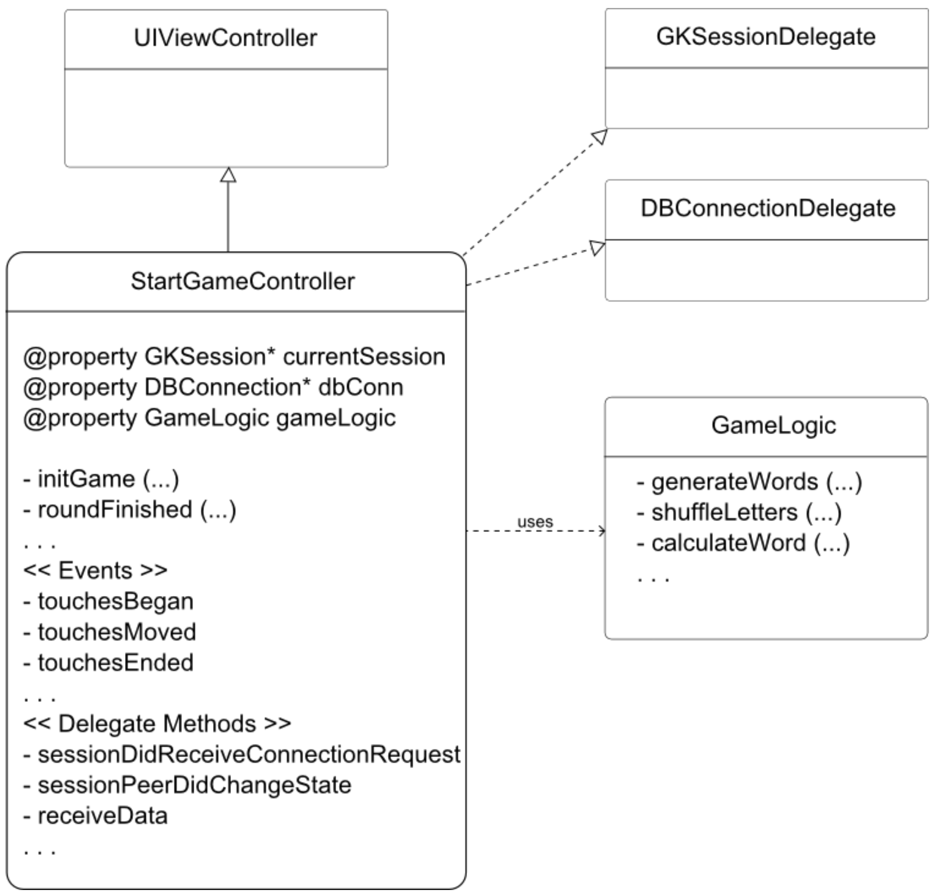


Figure 19 Class Diagram GameViewController

GameViewController is the main class of the application and certainly the most extensive one. The following section contains a step-by-step walkthrough of the class from the point it is loaded.

5.5.1.2 Implementation Details

When a new game or new round of a game is initiated, the device that acts as server will be responsible for:

- The process of generating words for all players.
    - *Drawing words.* In the XML file of the wordpool exists an element called “priority” for each item. Words (or rather word/question pairs) according to the number of players are drawn semi-randomly out of the wordpool. Words with higher priority have a higher change of being drawn.
    - *Calculate the missing letters.* The longer the word, the more letters will be removed.
    - *Shuffle the letters.* A certain number of letters will be spread among the other players. How many is determined by the length of the word-to-guess.
- Send the whole word, the incomplete word and the shuffled missing letters to the corresponding peer.
- Start a new round. The server will be notified when a client completes his word. When all players including the server have finished their words, a new round of the game will be initiated. The server will upload which words the players have finished (if they are signed in).

The client waits until it receives a word (and question, etc.) from the server, which will trigger a new game/round. When the word is completed, a “finished” message is sent to the server.

Although some details were left out here and there, this should cover all important functions. The only thing left that might be worth mentioning is how the drag and drop movement of letters and icons was implemented.

All moveable elements (views and image views) of the screen were placed in an array. When the user first touches the screen, the event “touchesBegan” is triggered and as a parameter the coordinates of the touch point are passed along. If the touch point is located within one of the frames of moveable elements, the object to move is successfully identified and can now update its position whenever “touchesMoved” is called. The event “touchesEnded” signals the release of the screen (or object).

5.6 Debugging

A proper testing and debugging environment is one of the most important features of an IDE. Xcode provides standard debugging features as well as a number of tools that can help to identify memory leaks or other performance issues. In addition, the iPhone/iPad Simulator enables developers to test their applications on desktop computers. The simulator is incredibly fast and will work without delay on notebooks such as the Macbook Air with limited resources. While more than adequate for most applications, several problems appeared due to the nature of the app:

- Debugging applications that are constantly communicating and maintaining a connection to other devices is always a challenge. Standard debugging procedures (break points, etc.) might not work.
- In a few cases, the simulator showed different behaviour than the real device.
- Only one instance of the simulator can be executed at the same time.
- The process of deploying apps on real devices was extremely tedious due to the restrictions of the student developer account.

The main problem was the lack of physical devices. To sufficiently test the application, at least 4 or 5 devices would have been required. Multiple instances of the simulator would at least partially solve the problem, but, as mentioned before, only one instance of the simulator can be executed at the same time. Interestingly, no proper workaround existed at the time to address this matter, at least not for the current versions of Xcode. The next logical step would be to install Mac OS X on another device (or virtual machine), install Xcode and start another simulator. However, this is not so easily accomplished because, by default, OS X cannot be installed on devices other than computers produced by Apple. Numerous workarounds exist to bypass that restriction, in form of manipulated versions of OS X that can be installed on real computers as well as modifications to existing virtual machine software that would make an installation possible. All of those solutions are heavily dependent on the hardware of the device. Plus, installing a manipulated version of OS X usually results in the problem that it cannot be updated, thus the current version of XCode cannot be installed. Suffice it to say that it took several days of experimenting to figure out a way to install a legal version of OS X on a virtual machine running under Windows 7 that would properly work. It should be noted that it is possible to install OS X on a virtual machine running in OS X. However, a minimum of 4 GB of RAM is required to launch the VM, which has not been available on the Macbook Air used for developing.

Another matter was the process of app deployment on physical devices. Apps can only be installed on a device with a valid provisioning profile for that device. Under normal circumstances, only a developer account is needed to acquire such a profile. In case of a student developer account, the process is the following:

Get the device ID, inform your supervisor that you would like to add the device for developing purposes, wait for the supervisor to provide the new provisioning profile, apply the profile to the new device, install the app. This drawn-out procedure makes it impossible to quickly grab a device for testing purposes.

This part concludes the discussion of the app “Buchstaben Post”. The next chapter will explain the web-interface, starting with the design of the database.

6 Design and Implementation of the Web-Interface

The web-interface consists of three parts, the website itself, the database and an interface to the user management.

6.1 Description of the Database

MySQL was employed as a database management system. The following graphic illustrates the database schema with all its relations.

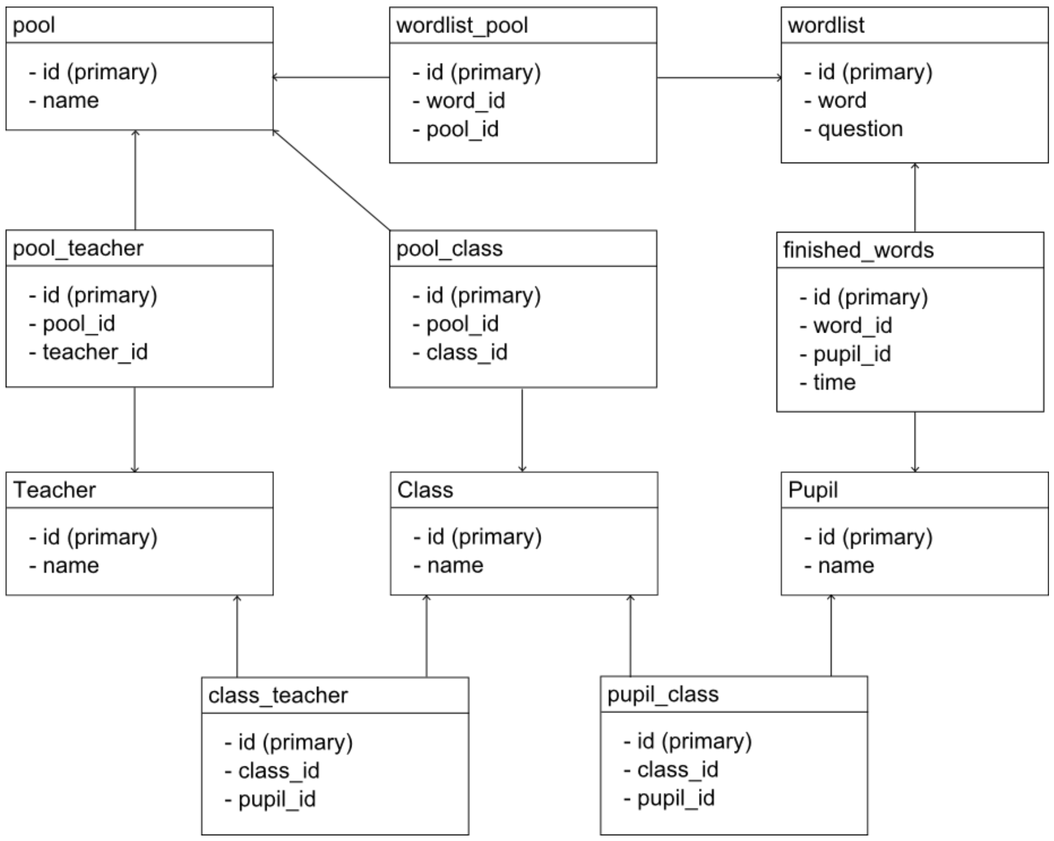


Figure 20 Database Schema

Teachers have classes and pools. A pool contains words from the wordlist, and is assigned to a class. For each pupil a list of finished words and the time it took to complete one round of the game is stored.

6.2 Description of the User Management System

The user manager is a database of local schools, containing accounts of teachers, pupils and their respective school classes. The system provides an interface for applications in form of a SOAP[31] web service.

“SOAP Version 1.2 is a lightweight protocol intended for exchanging structured information in a decentralized, distributed environment.” (W3C, 2007)

It is a W3C[32] recommended protocol specification that can be used for remote procedure calls or to exchange data between systems and relies on XML as its message format and Application Layer protocols like HTTP for message negotiation and transmission.

The protocol consists of three parts, an envelope, which defines what is in the message and how to process it, a set of encoding rules for data types and a convention for representing procedure calls and responses[33].

SOAP can be used in combination with WSDL[34] and an XML Schema to provide web services over the internet[35].

As the name suggests WSDL is an XML-based language for describing the functionality offered by a web service, or in more general terms, a format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information (Christensen, et al., 2001).

PHP provides a client for SOAP servers which can be used in WSDL or non-WSDL mode[36].

[31] Simple Object Access Protocoll (SOAP)

[32] World Wide Web Consortium (W3C)

[33] http://en.wikipedia.org/wiki/SOAP (Last visited: Nov. 2013)

[34] Web Service Description Language (WSDL)

[35] http://en.wikipedia.org/wiki/Web_Services_Description_Language (Last visited: Nov. 2013)

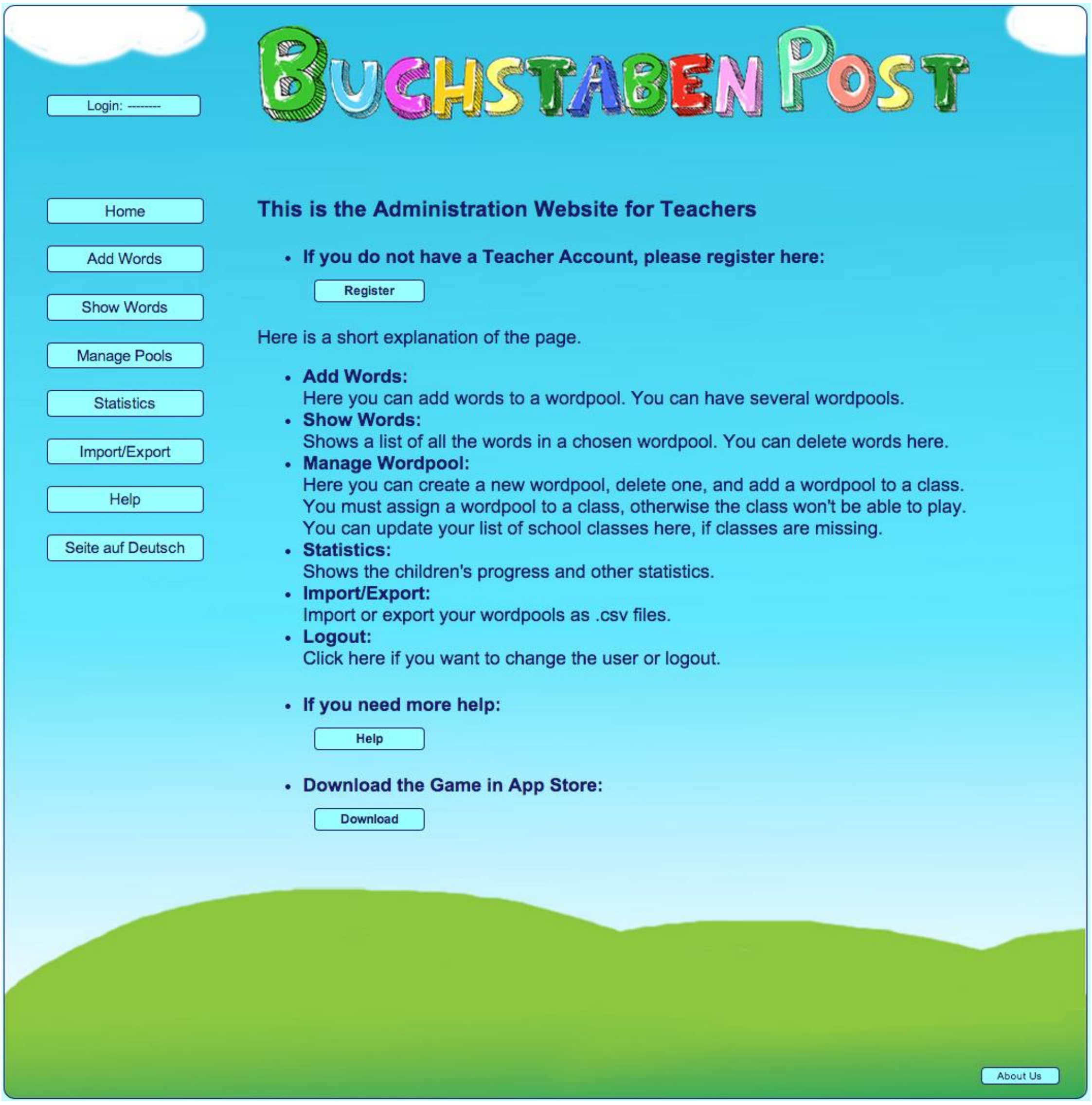
[36] http://php.net/manual/de/class.soapclient.php (Last visited: Nov. 2013)

6.3 Description of the Website

Following the same principle as in the description of the iPad application, the website’s user interface and features will be explained first. Afterwards the details of the implementation will be discussed.

6.3.1 User Interface and Features

The design of the website is simple and functional.



This is the Administration Website for Teachers

- **If you do not have a Teacher Account, please register here:**

Here is a short explanation of the page.

- **Add Words:**
  Here you can add words to a wordpool. You can have several wordpools.
- **Show Words:**
  Shows a list of all the words in a chosen wordpool. You can delete words here.
- **Manage Wordpool:**
  Here you can create a new wordpool, delete one, and add a wordpool to a class.
  You must assign a wordpool to a class, otherwise the class won't be able to play.
  You can update your list of school classes here, if classes are missing.
- **Statistics:**
  Shows the children's progress and other statistics.
- **Import/Export:**
  Import or export your wordpools as .csv files.
- **Logout:**
  Click here if you want to change the user or logout.

- **If you need more help:**

- **Download the Game in App Store:**

Figure 21 Website

When the page is loaded, the “Home” section is displayed, which gives an overview of the websites functions, explaining each site. When the user clicks on one of the links in the sidebar on the left, he/she will be asked to sign in. To login, a teacher account at the user management system is required. Once signed in, the field in the top left corner of the website will display the login status. To sign out or change the user, the “Logout” button has to be pressed.

The functionality of each page or section will be outlined with a few sentences:

- *Home.* This page offers quick help and introduces each section of the homepage. If a teacher does not have an account at the usermanager, a link provides access to the registration page of the usermangers website. A download link to the app store is provided as well.
- *Add Words.* In this section, teachers can add words to a wordpool. A word and question have to be typed in. Once a number of word/question pairs have been entered, the user has to press “Submit” and the words will be uploaded to the database.
- *Show Words.* The user can select a wordpool and the corresponding words will be displayed in a table view. Words can be removed by selecting a checkbox for the corresponding word. Once all words have been marked, they can be deleted by pressing the “Delete Words” button.
- *Manage Pools.* As the name implies, this part of the website deals with the management of wordpools. Wordpools can be created or deleted. A wordpool has to be assigned to a school class in order to be used. If no wordpool is assigned to a class, the class won’t be able to play when they are signed in. They are still able to play offline, though.
- *Statistics.* Shows how the children have performed while playing/learning with the app. For every pupil of the selected class, the time it took to complete a round of the game and the number of words he has finished is displayed.
- *Import / Export.* Every teacher has his own wordpools. A teacher will not be able to see or manipulate the wordpools of other teachers. While this is a desired functionality (to avoid chaos, etc.), most likely teachers will be using similar lists of words. Besides, they should have the possibility to save the wordpool in case the user account is changed. For this reason an import/export function was implemented. The user can download the selected wordpool in form of a .csv file and .csv files of the same structure can be imported. The files can be viewed and manipulated with every text editor that can process .csv files, for example, Microsoft Excel.

- *Help.* This section offers help about the use of the website as well as the iPad app. Besides a textual explanation, a video tutorial of both the website and the game was added.
- *View Page in English/German.* Besides the international version of the website in English, a local version in German Language exists as well.
- *Logout.* Displayed only when logged in, the user is able to logout or change the user if needed.

6.3.2 Technical Framework

The website was implemented in HTML, Javascript and PHP, using a PHP framework called CodeIgniter[37].

- HTML or HyperText Markup Language is used to structure and display content like text or images in a web browser. It is the main markup language for creating web pages.
- PHP, now standing for Hypertext Preprocessor, is a server-side scripting language designed for web development.
- Javascript is a scripting language that allows client-side scripts to interact with the user, control the browser or to alter the content of the document.
- CodeIgniter is an easy to use, lightweight web application framework to build dynamic web sites in PHP. Besides providing a rich set of libraries for commonly needed tasks it is based on the popular Model-View-Controller pattern[38], which will be explained in more detail.

6.3.3 The Model-View-Controller Pattern

The Model-View-Controller (MVC) pattern is a software design pattern that divides an application into three kinds of components, the controller, model and view, thus separating the representation of information from the user's interaction (Krasner & Pope, 1988).

[37] http://ellislab.com/codeigniter (Last visited: Nov. 2013)

[38] http://ellislab.com/codeigniter/user-guide/overview/at_a_glance.html (Last visited: Nov. 2013)

The graphic illustrates the pattern in its simplest form. A view requests information from the model that it needs for generating an output, the controller sends commands to update the model's state. The model notifies its associated views and controllers when there has been a change in its state.

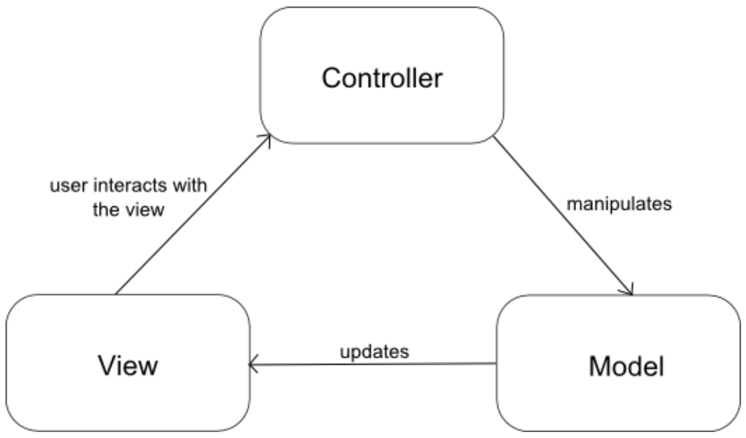


Figure 22 MVC Pattern

The modularity of components facilitates reuse and maintainability.

6.3.4 Design and Architecture

CodeIgniter is loosely based on the MVC pattern. Controller and views are necessary parts of a website, the model is optional. Regarding communication between the three components, the implementation of this website deviates a bit from the standard pattern, as depicted in the figure below.

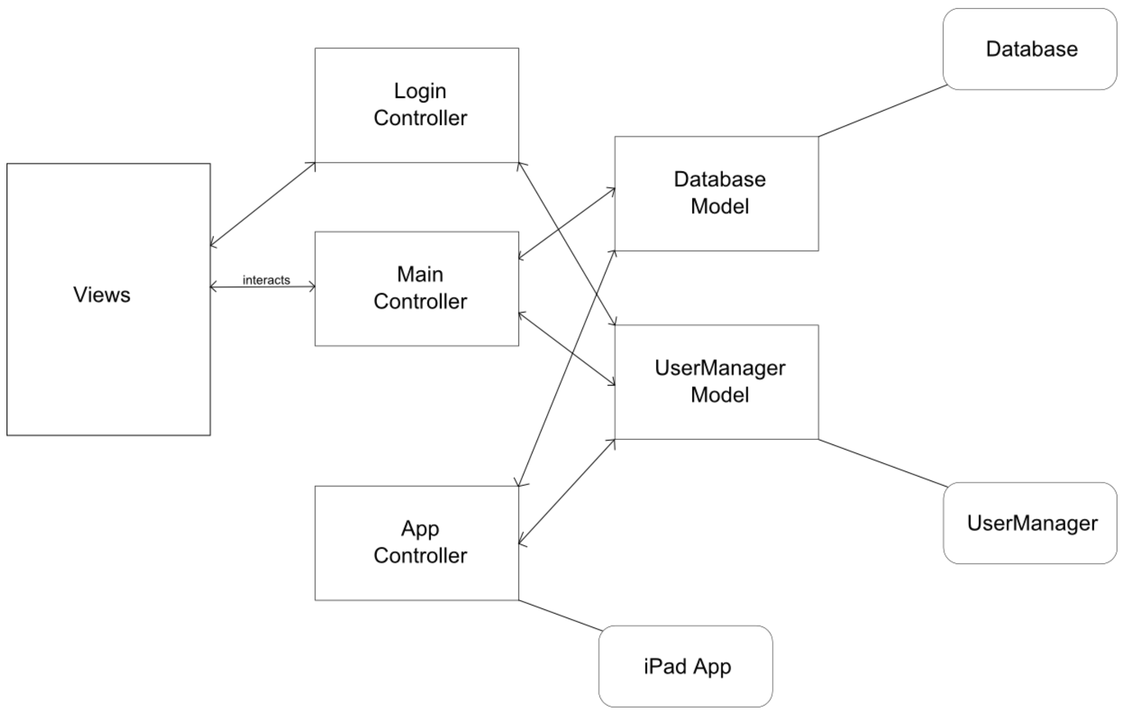


Figure 23 Website Architecture

As intended by the MVC pattern, the views of the website contain all the code for the presentation of the web page. There is a “login view” which displays a sign in form, a “show words view” which displays the corresponding section of the page, and so on.

When the user requests or submits data, the view will post a request to the assigned controller. The controller decides which methods to call from the model and passes along the data. The Login Controller is responsible for the login process only. The Main Controller handles all interaction with the website and the App Controller represents the interface to the iPad app.

The Database Model handles communication with the MySql database and contains all SQL queries. The Usermanager Model connects to the SOAP webservice of the Usermanager. After retrieving data from the database or webservice, the model passes it along to the controller and the controller will in turn update the views.

All code regarding the business logic of the website is implemented in the controller, the models only contains the functions necessary to communicate with either endpoint. This way, the website is easier to maintain. Most likely, when changes are required, they will only affect the SQL queries or communication with the Usermanager and not the logic of the website.

6.3.4.1 Security

Security is always an important issue regarding web applications. This section explains the mechanisms that were put into place to provide a sufficient measure of security.

The login process works as following:

- After username and password are entered, a SHA-256 hash of the password is generated. The credentials (username, password, and an additional string) are forwarded to the Usermanager for verification. The additional string contains a hash of the username, password, app ID and app key for verification purposes.
- In case of a successful authentication, the Usermanager will return a message containing the ID, role (student, teacher) and name of the user. The message can be verified using the public key of the Usermanager.
- The login status of the user is stored in the user's session variable and is encrypted. CodeIgniter does not utilize native PHP sessions, it generates its own session data. Session data can be encrypted, making the data highly secure and impervious to manipulation[39].

The login procedure from the iPad app follows the same principle and generates a SHA-256 hash of the password.

SQL injections are another matter of concern. SQL injections are insertions of malicious code into an entry field for execution. To prevent this from happening, input strings should be escaped, which will remove symbols that could be interpreted as SQL code. CodeIgniter provides its own string escape function[40]. Every query input is escaped in that matter.

Last but not least, the database contains as little information of the user as possible. Only user ID and username are stored. The password, real name and other personal data remains at the Usermanager.

[39] http://ellislab.com/codeigniter%20/user-guide/libraries/sessions.html (Last visited: Nov. 2013)

[40] http://ellislab.com/codeigniter/user-guide/database/queries.html (Last visited: Nov. 2013)

6.3.4.2 Maintainability

The website should be easily maintainable, which is one of the reasons why CodeIgniter is used. The MVC pattern greatly improves maintainability. Furthermore, CodeIgniter provides config files where all information regarding the configuration of the website and database is stored and easily accessible.

7 Using the App: Results and Discussion of the Trial

The field study took place at two primary schools in Graz. Both schools have so called “iPad classes”, in which every pupil is in possession of his/her own device. The first trial was performed at primary school Hirten, the second one at University of Teacher Education Styria. The field study took place in second grade classes, composed of 15 and 18 pupils. The children are used to working with the device and are proficient for their age. The trial was conducted as a participatory observation, followed by an interview with a group of children.

7.1 The Setting of the Test

The app was installed beforehand to save time. Every child had his own iPad in front of them. The players of each team were either sitting side by side or in front of each other. The test was split into two phases. First, the game was played in groups of two, afterwards in groups of four. Due to the nature of the game, a four player setting is more challenging and might have been too confusing for the beginning.

7.2 Results

The main part of the field study was the participatory observation. After a short explanation, the children could play the game for about an hour, while the teachers provided help when they were stuck or had questions.

Here is an overview of the positive and negative aspects (in that order) of the trial:

- The children quickly understood the concept of the game.
- While having some problems with the login and connection process, the game itself (game screen) was easy to use.
- There was steady communication between the players. This had been one of the main goals, but as a result the noise in the classroom was above the usual level, especially when the game was played in groups of four. While not necessarily negative, it is an aspect that has to be considered when employing the game.
- In almost every case, the children cooperated very well. When one player could not finish a word, he/she got help from his teammate. Or he/she tried to help his/her partner to finish his/her word and would then receive help him/herself. Some of the children were actively helping or grabbing letters by reaching out to their neighbor's iPad.
- The children generally seemed to enjoy the game and the resulting team-work.
- They were motivated by finishing a word or round of the game.
- Sometimes words were challenging or unknown to the children or the hint or question was too vague. Yet they were very flexible in finding solutions, discussing among themselves and only asking for help when they could not reach a conclusion.
- Some groups were more competitive than others, trying to solve the puz-zle as quickly as they could. One group even made a contest who could finish his/her word first.
- Another aspect worth mentioning is that the children actually tried to guess the word by reading the hints, and not just by looking at the word itself or just randomly trying out letters. This is to some degree due to the concept of the game. Since the letters were spread among the players, in order to ask for specific letters, they had to guess the word correctly. Nevertheless, sometimes they would resort to random trying, but this was mostly the case when the question or hint was too vague.
- The main problem was of technical nature and appeared in form of con-nection failures as a result of establishing a connection over Bluetooth. The devices had both Bluetooth and WLAN activated (in case WLAN would not work due to network restrictions). In previous tests with only a limited number of devices (a maximum of four) these settings seemed to work well. Somehow, the Bluetooth framework appeared to be unable to

cope with a multitude of connection requests, cancelling each other out. Disabling Bluetooth on all devices fixed the problem.

- The connection process posed some more difficulties. Due to the eagerness of the children to play, some of them were connecting to random players (instead of the one they were sitting next to), while others would not wait long enough for their partners to find the game or until it appeared in the in the list of available games (about 1-5 seconds delay). This, in combination with the technical problem mentioned above, lead to some confusion in the first minutes of the trial.
- As explained before, players have to sign in to receive custom wordlists. Filling in the correct password (a combination of letters and numbers, about 8-10 symbols long) was a challenge for some of the children, and they had to try several times.
- For a few pupils the difficulty was too high. They had trouble in reading and understanding the questions and spelling the words.
- In some cases children found it difficult to place the letter at the right position, because their dragging movement was not precise enough.

After observing the children's performance, a short interview with a handful of pupils, one of each group of players, was conducted. For this purpose, a number of questions and statements had been prepared. The children were asked to discuss the questions by themselves. Then, as a group, they had to show us if they agreed with a statement or not (by putting down smileys ranging from happy to sad). The prepared questions and answers are summed up in the following table:

Table 2 Interview Results

Question	**Answer**[41]
Did you like to play together?	1
Was the app easy to use?	1-2
Did you learn new words?	1
Was the difficulty of the game (words) ok?	1-3
Would you like to play again?	1

[41] “1“ equals a very happy smiley, “5” a sad smiley.

Discussion of the interview:

- *Question 1.* Playing and learning together is the main goal of the game. All children agreed that it was fun to do so.
- *Question 2.* Most of the children found it easy to use, some of them were a bit unsure. As a group, the answer was 1, however, after asking if there were any problems, some said that it was a bit confusing to start a game and find players. To some degree this was the result of problems with Bluetooth cancelling out connections in the first trial.
- *Question 3.* The wordlist contained a few words previously unknown to the children. In most cases, they were able to guess the new word by themselves.
- *Question 4.* The opinions of the children differed somewhat about this point, ranging from easy to sometimes difficult. This was mainly due to two reasons: First, the wordlist used for the tests was the standard wordlist of the app, and not customized for the school classes' curriculum. Therefore, some words were unknown or difficult. Second, one of the school classes consisted of non-native German-speaking children, and their state of knowledge differed from one pupil to another. Surprisingly, this did not create any problems and enriched the testing environment, because it demonstrated how well the children were working together. Pupils with better reading and writing skills were supporting the other players.
- *Question 5.* The children all agreed that they would like to play again soon.

Afterwards, the children were asked to talk about their experience with the game. The discussion affirmed the results of the observation, the children saying that they enjoyed playing together and had a lot fun. Some of them found the words a bit difficult. Among the most prominent statements were "We had fun" and "We would like to play some more". About a week after the test, further feedback was received from attending teacher, stating that the children chose to play "Buchstaben Post" (out of other learning games and activities) during their "free work time".

7.3 Discussion

The project is focussed on cooperation among peers and how to incorporate new technology and gaming into a classroom setting. Regarding this area of expertise, first field studies rendered practical feedback and the basis for further assumptions.

In summary it can be said that:

- The app fostered communication and collaboration.
- Given the right setting, children need little to no encouragement to support each other and work together.
- The presentation of learning content as a game is more enjoyable for children and motivates them to “play again”.
- There are still a few technical difficulties that need to be addressed.

While the encountered problems were of minor nature and can easily be fixed, future work and more sophisticated studies need to be conducted regarding the learning results of the game compared to pure learning applications or traditional learning schemes.

8 Conclusion

The combination of wireless network technology and mobile devices has opened countless new ways to deliver content and exchange data. In the personal sector, those technologies (smartphones, tablets, etc.) have become omnipresent. Beyond private use, the potential for schools and educational purposes is incredible. Nevertheless, due to a number of reasons (missing experience, cost, etc.) mobile devices are still rarely employed, and customized learning applications for schools are limited.

8.1 Project Summary

Focusing on the aspect of cooperation and collaborative learning, the goal of this project is to use digital devices to connect learners to strongly assist the communication between peers. The fundamental idea was to develop an application where learners actively engage into collaborative work.

The realization of the project consists of two parts.

- The development of a prototype application and website. The application is a cooperative multiplayer learning game for iPads, the website an interface for teachers to create custom content and to monitor the children's performance.
- A field study to test and evaluate the prototype. The trials took place at two primary schools in Graz where the app was tested in a proper classroom setting.

The results of the field study look promising. The app supports and actively enables cooperative behaviour and collaborative learning. Children are highly motivated to use this kind of application and they enjoy the resulting teamwork.

8.2 Lessons Learned

Developing applications for children can be a challenging task. Especially the design of the user interface requires a lot of care and attention to details. While today's generation of school children is familiar with mobile devices and quick to grasp the concepts of an application, their approach to new applications is a fundamentally different one than that of adults. Where adults are cautious and deliberate, children are curious and eager to begin. This lead to two insights:

- Children are quick to discover all kinds of functionality.
- Complex procedures pose a challenge. Children are unlikely to read or listen to advice before trying something out.

On the software side, several things are worth mentioning:

- With XCode and the Cocoa framework, Apple provides a rich developing environment, including a lot of features and libraries that help considerably in the development of an application.
- While the restrictive policy of the operating system enhances stability and security, it also limits programmer's options in dealing with problems.
- XCode's ability to test applications which connect several devices is severely limited due to the fact that only one instance of the simulator can be executed at a time.

8.3 Concluding Remarks

The project is focussed on cooperation among peers and how to incorporate new technology and gaming into a classroom setting. First field tests rendered practical results and it can be stated that this study is a step towards collaboration through mobile devices.

Certainly, this field of research has much to offer. Regarding the project, several things come to mind:

Further studies have to be performed, especially concerning the learning results of the application and how it compares to conventional learning methods. Considering the social aspect, several questions arise:

Does cooperation during the game affect the social behaviour in the classroom? Does it have a positive influence on the children and the class? Does it help them to work together in general, and not only when using the application?

From a technological point of view, the possibilities are numerous. To begin with, the concept of the game could be expanded. As the field study demonstrated, children very much like playing or learning together. Tablets or similar devices make it possible to sit together and communicate in a direct manner. Building on that concept, a number of learning games or applications could be developed that focus on collaborative learning and social interaction. Also, the number of connected devices is not limited in any way. It is possible to connect more or even all devices in a classroom. An assignment could be split into smaller parts, and while the whole class works on finishing that assignment, groups of children can pick one of several tasks and work together in solving it.

The possibilities are numerous and fascinating and with the rapid advancement of mobile technologies, more and more ways to enhance the learning experience will be possible to realize - not only of school children, but for learners of every kind and age.

References

Ally, M. (2009). Mobile Learning Transforming the Delivery of Education and Training. *Issues in Distance Education series*, pp. 2;29.

Budiu, R., & Nielson, J. (2010). *Usability of Websites for Children: Design Guidelines for Targeting Users Ages 3-12 Years.* Nielson Norman Group Report.

Chan, T. W., Roschelle, J., HSI, S., Sharples, M., BrownT, Patton, C., et al. (2006). One-to-One Technology-Enhanced Learning: An Opportunity for Global Research Collaboration. *Res. Practice Tech. Enhanced Learning, 1*(3).

Chiu, M. M. (2000). Group Problem-Solving Process: Social Interactions and Individual Actions. *Journal for the Theory of Social Behaviour, 1*, pp. 27-49.

Christensen, E., Curbera, F., Meredith, G., & Weerawarana, S. (20011). *Web Services Description Language (WSDL) 1.1.* Accessed November 2013 at http://www.w3.org/TR/wsdl

Consortium, W. W. (2008). *Extensible Markup Language (XML) 1.0 (Fifth Edition).* Accessed November 2013 at http://www.w3.org/TR/REC-xml/

Dawes, L. (2011). *Shaping the Future - Realising the potential of informal learning through mobile.* http://www.gsma.com/mobilefordevelopment/wp-content/uploads/2012/05/mLearning_Report_230512_V2.pdf: GSMA, Mastercard Foundation.

Dewey, J. (1916). *Democracy and Education. An introduction to the philosophy of education (Reprint 1997).* Rockland (NY): Free Press.

Dillenbourg, P. (1999). *Collaborative Learning: Cognitive and Computational Approaches. Advances in Learning and Instruction Series.* New York, NY: Elsevier Science, Inc.

Ebner, M. (2010). *iPad Human Interface Guidelines.* Accessed November 2013 at http://itunes.tugraz.at/media/items/iphone_application_development-apple_ttt-2010-08/1281959367-hci_ipad.pdf

Ebner, M., & Holzinger, A. (2007). Successful implementation of user centered game based learning in higher education: An example from civil engineering. *Computers & Education, 3*(49), pp. 873-890.

Ebner, M., Böckle, M., & Schön, M. (2011). Game Based Learning in Secondary Education: Geographical Knowledge of Austria. *World Conference on Educational Multimedia, Hypermedia and Telecommunications*, (pp. 1510-1515).

Ebner, M., Kolbitsch, J., & Stickel, C. (2010). iPhone / iPad Human Interface Design. *Human-Computer Interaction in Work & Learning, Life & Leisure*, pp. 489-492.

Ebner, M., Schön, S., Khalil, H., & Zuliani, B. (2013). Cooperative Face-to-Face Learning with connected Mobile Devices - The Future of Classroom Learning?

Egenfeldt-Nielson, S. (2011). *Beyond Edutainment: Exploring the Educational Potential of Computer Games.* lulu.com.

Erikson, E. (1963). *Children and society.* WW Norton & Company.

Facer, K. (2003). *Computer games and learning.* Accessed November 2013 at http://admin.futurelab.org.uk/resources/documents/discussion_papers/Computer_Games_and_Learning_discpaper.pdf

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). Design Patterns. Elements of Reusable Object-Oriented Software.

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). Design Patterns. Elements of Reusable Object-Oriented Software.

Gong, J., & Tarasewich, P. (2004). Guidelines for handheld mobile device interface design. *In Proceedings of the 2004 DSI Annual Meeting.*

Gossen, T., Nitsche, M., & Nürnberger, A. (2012). Search user interface design for children: challanges and solutions. *EuroHCIR2012.*

Grimus, M., & Ebner, M. (2013). M-Learning in Sub Saharan Africa Context - What is it about. *Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 2013*, (pp. 2028-2033).

Grimus, M., Ebner, M., & Holzinger, A. (2013). Mobile Learning as a chance to enhance education in developing countries - on the example of Ghana. *mLearn 2012 Conference Proceedings*, *955*, pp. 340-345.

Hackfort, D. (2003). *Studientext Entwicklungspsychologie 1: Theoretisches Bezugssystem, Funktionsbereiche, Interventionsmögilchkeiten.* Vandenhoeck & Ruprecht.

Hannak, C., Pilz, M., & Ebner, M. (2012). Fun - A Prerequisite for Learning Games. *Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 2012* (pp. 1292-1299). Chesapeake, VA: AACE.

Holzinger, A. (2002). *Multimedia Basics. Volume 2: Cognitive Fundamentals of Multimedial Information Systems.* New Delhi: Laxmi-Publications, Available in German by Vogel Publishing.

Hourcade, J. P., Bederson, B. B., Druin, A., & Guimbretiere, F. (2004). Differences in pointing task performance between preschool children and adults using mice. *ACM Transactions on Computer-Human Interaction, 11*(4), pp. 357-386.

Huber, S., & Ebner, M. (2013). iPad Human Interface Guidelines for M--Learning. In *Handbook of mobile learning* (pp. 318-328). Z.L. Berge and L.Y. Muilenburg (Eds.).

Hutschinson, H., Druin, A., Bederson, B. B., Reuter, K., Rose, A., & Weeks, A. C. (2005). How do I find blue books about dogs? The errors and frustrations of young digital library users. *Proceedings of the 11th International Conf. on Human-Computer Interaction (HCII 2005).* Mahwah, NJ: Lawrence Erlbaum Associates.

ITU. (2011). *Statistics of the International Telecommunications Union: Measuring the Information Society.* http://www.itu.int/ITU-D/ict/publications/idi/index.html.

Johnson, D. W., & Johnson, R. T. (1999). Making Cooperative Learning Work. *Theory Into Practice, 38*(2), pp. 67-73.

Johnson, D. W., & Johnson, R. T. (2009). An Educational Psychology Success Story: Socal Interdependence Theory and Cooperative Learning. *Educational Researcher, 38*(5), pp. 365-379.

Johnson, D. W., Johnson, R. T., & Smith, K. A. (1991). *Cooperative Learning: Increasing College Faculity Instructional Productivity.* The George Washington University: School of Education and Human Development.

Jones, S. (2003). *Let the Games Begin. Gaming Technology and Entertainment among College Students.* WASHINGTON, D.C: http://www.pewinternet.org/.

Jordan, D. W., & Metais, J. (2006). Social skilling through cooperative learning. *Educational Research, 39*(1), pp. 3-21.

Kafai, Y. B. (2001). *The Educational Potential of Electronic Games: From Games-To-Teach to Games-To-Learn*. Accessed November 2013 at http://culturalpolicy.uchicago.edu/papers/2001-video-games/kafai.html

Karlsson, P., & Djabri, F. (2002). Analogue Styled User Interfaces: An Exemplified Set of Principles Intended to Improve Aesthetic Qualitites in Use. *Proceedings of the 35th Hawaii International Conference on System Sciences.*

Kloper, E., Squire, K., & Jenkins, H. (2002). Environmental Detectives: PDAs as a Window into a Virtual Simulated World. *Proceedings of the IEEE International*

Workshop on Wireless and Mobile Technologies in Education (pp. 95-98). IEE Computer Society.

Krasner, G., & Pope, S. T. (1988). *A Description of the Model-View-Controller User Interface Paradigm in the Smalltalk-80 System.* ParcPlace Systems.

Large, A., Beheshti, J., & Rahman, T. (2002). Design criteria for children's Web portals: The users speak out. *Journal of the American Society for Information Science and Technology, 53*(2), pp. 79-94.

Laughlin, P. R., VanderStoep, S. W., & Hollingshead, A. B. (1991). Collective versus individual induction: Recognition of truth, rejection of error, and collective information processing. *Journal of Personality and Social Psychology, 61*, pp. 50-67.

Lee, K. (2013). *Pro Objective-C.* Apress.

Liang, J. K., Liu, T. C., Wang, H. Y., Chang, B., Deng, Y. C., Yang, J. C., et al. (2005). A few design perspectives on one-on-one digital classroom environment. *Journal of Computer Assisted Learning, 21*(3), pp. 181-189.

Malone, T. W. (1980). What makes things fun to learn? Heuristics for designing instructional computer games. *Proceedings of the 3rd ACM SIGSMALL Symposium and the 1st SIGPC Symposium on small systems*, (pp. 162-169).

Mann, B. D., Eidelson, B. M., Fukuchi, S. G., Nissman, S. A., Robertson, S., & Jardins, L. (2002). The development of an interactive game-based tool for learning surgical management algorithms via computer. *The American Journal of Surgery, 183*(3), pp. 305-308.

Moore, G. A. (1999). *Crossing the Chasm, Marketting and Selling High-Tech Products to Mainstream Customer (revised edition).* New York: HarperCollins Publishers.

Motschnig-Pitrik, R., & Holzinger , A. (2002). Student-Centered Teachings Meets New Media: Concept and Case Study. *IEEE Journal of Technology & Society, 5*, pp. 160-172.

Murphy, P. K., Wilkinson, I. A., Soter, A. O., Hennessey, M. N., & Alexander, J. F. (2009). Examining the Effects of Classroom Discussion on Students' Comprehension of Text: A Meta-Analysis. *Journal of Educational Psychology, 3*, pp. 740-764.

Naidu, S. (2005). Evaluation the usability of educational websites for children. *Usability News, 7*(2).

Norman, D. (2004). *Emotional Design.* New York: Basic Books.

Oblinger, D. (2004). The next generation of educational engagement. *Journal of Interactive Media in Education*, pp. 1-18.

O'Reilly, T. (2007). What is Web 2.0?: Design Patterns and Business Models for the Next Generation Software. *Communications & Strategies, 65*, pp. 17-37.

Papastergiou, M. (2009). Digital Game-Based Learning in high school Computer Science eduction: Impact on educational effectiveness and student motivation. *Computers & Education, 52*(1), pp. 1-12.

Perez, C. (2002). *Technological revolution and financial capital - the dynamic bubbles and golden ages.* UK: Edward Elgar Publishing Limited.

Piaget, J., Inhelder, B., & Inhelder, B. (1969). *The psychologie of the child.* Basic Books.

Pinkwart, N., Hoppe, U., Milrad, M., & Perez, J. (2003). Educational scenarios for cooperative use of personal devices. *Journal of Computer Assisted Learning, 19*(3), pp. 383-391.

Poupyrev, I., Maruyama, S., & Rekimoto, J. (2002). Ambient Touch: Designing Tactile Interfaces for Handheld Devices. *Proceedings of the 15th annual ACM symposium on User interface software and technology*, (pp. 51-60).

Prince, M. (2004). Does Active Learning Work? A Review of the Research. *Journal of Engineering Education, 93*(3), pp. 223-231.

Raines, C. (2002). *Managing Millennials.* Accessed November 2013 at http://www.generationsatwork.com/articles_millenials.php

Rogers, E. M. (1995). *Diffusions of innovation.* New York: The Free Press.

Schneiderman, B. (1998). *Designing the User Interface - Strategies for Effective Human-Computer Interaction.* Addison Wesley.

Smith, K. (1979). Learning Together and Alone: Cooperation, Competition, and Individalization. *25th Annual NACTA Conference*, (pp. 23-26). University of Minnesota, St. Paul.

Solomon, P. (1993). Children's information retrieval behavior: A case analysis of an OPAC. *Journal of the American Society for Information Science*(44), pp. 245-264.

Stuart, A. (2007). *When should kids learn to read, write, and do math?* Accessed November 2013 at http://children.webmd.com/features/when-should-kids-learn-read-write-math

Vygotsky, L. (1978). *Mind in society: The development of higher psychological processes.* Cambridge: Harvard University Press.

W3C. (2007). *SOAP Version 1.2 Part 1: Messaging Framework (Second Edition).* Accessed November 2013 at http://www.w3.org/TR/soap12-part1/

W3C. (2008). *Extensible Markup Language (XML) 1.0 (Fifth Edition).*

Zechner, J., & Ebner, M. (2011). Playing a Game in Civil Engineering. *14th International Conference on Interactive Collaborative Learning (ICL2011)– 11th International Conference Virtual University (vu'11*, (pp. 417-422).

Zurita, G., & Nussbaum, M. (2004). Computer supported collaborative learning using wirelessly interconnected handheld computers. *Computers & Education, 42*, pp. 289-314.