university of groningen

university college groningen

jIdeaGenerator

A Novel Approach to Idea Generation in Statically-Typed Languages

Bachelor Thesis

Happy van Der Promovendum

Faculty of Mathematics and Natural Sciences
University of Groningen

31. December 9999

Supervisor:

Dr. Mircea Lungu

Abstract

Learning has changed over the years. People do not learn only from books anymore. Although Books are still used, other tools for learning are gaining popularity. Websites that can test the knowledge about words of another language of the user are quite common and they have improved themselves over the years.

There is however a disadvantage. The user still needs to find the time to sit down in front of the screen and focus to learn the words. With the upcoming market of smartwatches a new possibility for learning is created. A possibility that makes the user able to learn words whenever the user wants to.

The smartwatch is however a relatively new platform with a small screen and thus we had to investigate what is the best way to build an app for the smartwatch. And can an app for the smartwatch help to accelerate the memorization process of somebody who is learning the vocabulary of a second language?

The app is useful for busy people who can read and understand the language, but want to improve their vocabulary by reading articles in this language and selecting the words the user did not know. These words can be learned by the smartwatch app to improve their vocabulary. The smartwatch is a suitable solutation, because it is quickly accessible which makes it ideal for situations like waiting for and waiting in the elevator or waiting for the bus or other situations where time is lost by waiting. The user only has to twist his or her wrist to continue learning. This is the reason why it is more convenient to use than an app for the smartphone. It is much more easier to access the app on the watch than on a smartphone.

Applications that are built with the same motivation as our app are mostly built for a smartphone like an Android wallpaper. Our approach is different, because we are going to try to use a smartwatch.

Several users have used the app for five days and they provided us with results and feedback for our research.

Contents

1 Introduction

The way people learn has changed over the years. The development of information and communication technology has lead to new ways of learning. People can now find fast and really specific information on the web and learn new things using ICT tools, like laptops, iPads and other electronic devices. The traditional book is being less and less used. We can see this in high schools where students learn their languages on the laptop with a rehearsing program. These rehearsing programs should accelerate the memorization process, so the student can work more efficiently and therefore has more time for other stuff and can work more efficiently.

Accelerating the memorization process of somebody who is learning the vocabulary of a second language is not only valuable for students, but also for other people who want to learn a second language for other reasons. But even with rehearsing programs learning a second language can be hard and time consuming thing to do. Since learning a second language is so time consuming, people often don't have enough time for it. They are busy with other important things like work and don't have the time to really learn every day. Although for these type of people it seems like they have no time to learn, but in fact the time they lose for waiting for the bus, taking the elevator and walking to the car could be used more efficiently. Using these little time units is called micro learning. In this thesis we are going to try to find out how we can fill these little time spots in the best way using a smartwatch.

The application for the smartwatch will be a watch face, this is the screen that the user will see when the screen is on. The main function of the watch face is to show the time, our application however offers the user a word from which the translation can be revealed by tapping on the screen. The user will be stimulated to think of the right translation before seeing it, after revealing it the user can give the watch feedback by pressing the green ('I had it right in my head') or red button ('I had it wrong in my head'). The algorithm used for displaying words is based on the way in which people learn with flashcards. This is to speed up the memorization process, wrong words will be repeated earlier then words which the user had right in his mind.

The app for the smart watch is part of Zeeguu, which is a research project designed to speed & fun up vocabulary learning in a new language. Its based on three fundamental principles: only read the stuff you like, have words everywhere with you and practice with personalized exercises. This basically means the users read articles they like, tap on the words they don't know and practice them later. The words will be saved in their accounts and can be accessed at any time. We build an app for the smartwatch, so users have an complementary tool for learning their words in the Zeeguu account and thus increase the memorization process of learning a second new language. In this thesis we will research what is the best way to build such an app, therefore a user study was designed to answer the following research question:

How do users use a smartwatch application to accelerate the memorization process of somebody who is learning the vocabulary of a second language?

Structure of this Thesis

- Related work: the general research field is described and how others tried to solve this solve problem.
- The design: the user interface is described, explaining the choices which have been made.
- The implementation: the code structure, algorithms and implementation choices are explained.
- Usage results: explaining the tests and a summary of the results with various diagrams.
- Evaluation: a discussion about why this research was successful.
- Conclusion and future work: conclusions about the usage results and describing possible future work.

2 Related Work

Describe the field in general and how others have tried to solve this problem.

Your goals are to:

- show you are aware of current state of knowledge (theoretical, methodological, applied) that relates to your research topic
- To indicate a gap/question worthy of investigation

(Thornbury. S.(2002) How to teach vocabulary. Harlow. Longman Pearson.) According to Scott Thornbury (2002) knowing a word involves knowing its form and its meaning. Knowing its form means that you know whether it is a noun, a verb or a preposition etc. and how to spell it. When focussing on its meaning you not only need to know what the word means in the L1, but also what register it is used in, what category of words it belongs to: for instance: fruit, animals, plants, transport or even to groups of abstract words. And you need to know how it is used in a sentence or what chunks or collocations it can be used in.

(Nation P. (2001). Learning Vocabulary In Another Language. Cambridge: Cambridge University Press.) given guidelines (Nation, 2001) in An Introduction to Applied Linguistics: 1. Retrieve rather than recognize; by studying with means of flashcards. Write the word on one side and write the translation on the other side. This will inflict retrieval after the first meeting. Each retrieval reinforces the connection between the form of the word and its meaning (Baddeley, 1990). When learners see the word and its meaning at the same time, this strengthening will not happen. 2. Learn about fifteen to twenty words at the same time. Difficult vocabulary should be learnt in small groups to allow more repetition and thoughtful processing. 3. Space the repetitions; this results in longer lasting learning. Spacing of about one hour will have more effect than on going repetitions. Also repeating after a day, a week and a month have a positive effect on the consolidating in the long-term memory. This spacing or spaced repetitions can also be called distributed practice. It is best to teach a new set of words and then only pick out the first couple of words. Then go back and test these, then present some more, then revisit the first words again and so on. When every word gets learnt better, the time between testing can be gradually extended. The idea is that vocabulary that was learnt the lesson before, should be revisited. The interval between successive tests should gradually be enlarged. 4. Repeat the words aloud or to oneself; this will create a good chance of consolidation into the long-term memory. 5. Process the words thoughtfully. If words are difficult to learn, use of depth processing techniques is recommended. Try to have as many associations with the word as you can. So the more decisions the learner makes about a word and demands greater cognitive thinking, the words will be remembered better. A good example of a higher order thinking skill (HOTS) is to define what kind of word it is, for example a noun or a verb. To use it in a complete sentence in a new situation would be the deepest way of consolidation of the word. 6. Avoid interference; words of similar spelling of meaning should not be learnt together. Interference makes learning more difficult. Which also means that for instance learning the days of the week in one time; is the wrong thing to do. Studying this it strikes us that the way we have pupils study their vocabulary at this moment, is actually learning words by category and according to this theory that is not a clever thing to do. 7. Avoid serial learning. You will have to change the order of the cards in your stack every now and then. Or else you will just know the order by heart, and one word will remind you of the meaning of another word. 8. Use context where this helps. Some words are most usefully learnt in a phrase or sentence. This especially applies to verbs. The word ”word” is a noun, adjective, and verb. Noun: ”The words on the page blurred as she moved the magnifying glass”. Verb: ”I word my sentences carefully, so as not to confuse you.” Adjective: ”I like word games.”

(Anderson, J.P. and Jordan, A.M. 1928. Learning and retention of Latin words and phrases. Journal of Educational Psychology 19. 485-496.) For example, Anderson and Jordan (1928) measured recall immediately after learning, after one week, after three weeks and after eight weeks. The percentages of material retained were 66%, 48%, 39% and 37% respectively. This indicates that the repetition of new items should occur very soon after they are first studied, before too much forgetting occurs. After this the repetitions can be spaced further apart.

(I.S.P. Nation, 1982. Beginning to Learn Foreign Vocabulary: A Review of the Research)

If there is a delay between the presentation of a word form and its meaning, learners have an opportunity to make an effort to guess the meaning, and presumably this extra effort will result in faster and longer retained learning. However, the guessing can only be successful if the foreign word form gives a good clue to its meaning, either because the foreign and native words are cognates, or because the word form and its translation have previously been seen together. Experimental evidence shows that simultaneous presentation of a word form and its meaning is best for the first encounter, and thereafter, delayed presentation is best because there is then the possibility of effort leading to successful guessing.

In order to grasp the full meaning of a word or phrase, students must be aware of the linguistic environment in which the word or phrase appears.

(Pradhan, D. and Sujatmiko, N. 2014. Can smartwatch help users save time by making processes efficient and easier?) Smartphone: how many step is required to do things? Given the phone has been picked out from pocket and now is on hands, a person still needs to do some steps. Typically the steps are: o Unlocking the phone. While in pocket, a phone is typically screen-locked to avoid unwanted use. Unlocking requires several taps or slides. o Finding app. A person typically has number of apps that is more than one phone screen can hold and therefore apps icons are placed in multiple home-screens. Hence finding the apps required another one or two taps or slides. o Running the app itself. This may vary based on how easy the app?s user interface is, but it definitely needs more than just two taps. With the number of steps above, in a mobile situation such as while driving, cycling, walking or running, the use of phone is not a convenient thing, time taking and could be even unsafe.

There is other time-taking aspects that people may see as disadvantage of smartphone. Smartphone is mobile but we need to carry, which implies while we are not carrying it, there could be a chance that we forget where we place the phone. Hence finding or searching for the misplaced phone is another time-taking exercise, compared to wearing thing.

3
The Design

Making an app for a smartwatch involves quite some thinking about the design. Compared to a smartphone the available display is circular and much smaller and thus designing and redesigning the layout with the time, the words, the buttons and some information components took some time. The app is a watchface and this comes with some restrictions when it comes to the different inputs the watch can receive from the user. An app that is set as a watchface can only detect touch, swipe up and swipe down. The app begins with a login screen and after entering a valid code the main screen is shown. All the decisions related to the designs of those two screens are described below.

3.1 Login screen

Before the user can use the app a code is needed that is used for receiving the user's words from the server. To get this code the app starts with a login screen. The login screen first consisted of ten buttons with the ten numbers, a clear button and a okay button. On the top of the screen there are four small rectangles for displaying the pressed numbers (see figure login_screen.png).

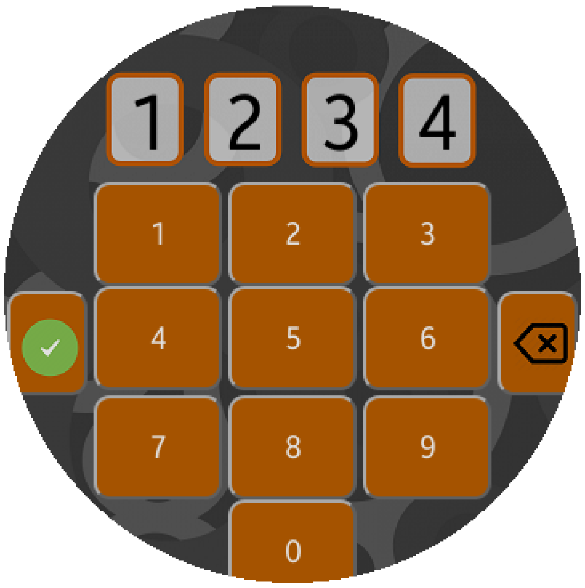

Figure 3.1: login screen v1.0

After some testing and discussions it was decided that the feedback of the pressed numbers on the top of the screen were to small and thus some rethinking was needed to come up with a new design. An idea was to make four rotating disks that was inspired from securing a travel suitcases.

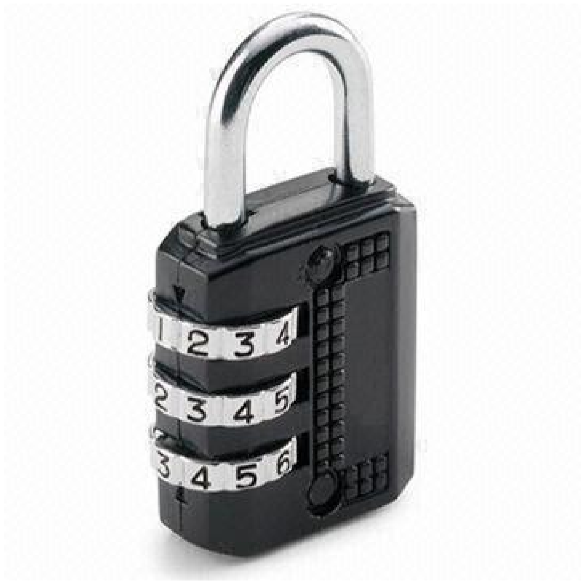

Figure 3.2: lock

The advantage is that no buttons for the numbers were needed so the numbers could be place in the middle. In the middle they can be displayed in a larger font since more space is available due to the circular shape. However, the detection of swipe events were insufficient for rotating the disks and thus the idea for rotating disks was changed to improve the functionality. The positions of the numbers stayed the same, but instead of swiping an increase and decrease button was added above and below each number (see figure login_screen_2.png).

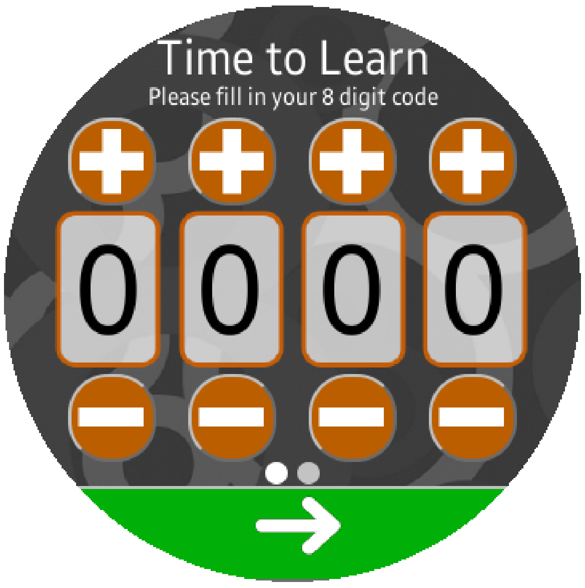

Figure 3.3: login screen v2.0

To increase the safety of a user's account it was decided to use a code of eight digits and therefore a page indicator was placed below the decrease buttons to notify the user that the user should insert 4 digits two times. On the bottom of the screen a next button is placed to go to the next page for inserting the second 4 digits of the code or to confirm that the 8 digits were inserted.

3.2 Main screen

When a valid code is inserted, the main screen becomes visible. During the project, the main screen had had different layouts. All the decisions are sorted in different categories: time, words, background, buttons, information components, option lists, profile and effects.

3.2.1 Time

An important part of the app is displaying the time, since the app will be the first thing the user will see when the watch screen turns on. This importance was not clear at the beginning of the project, therefore the first design contained the time in a small black font. The size of the font was based on the hill that was displayed on the background. With the chosen font the time fitted in the hill in the middle to increase the readability (see figure version 1.png).

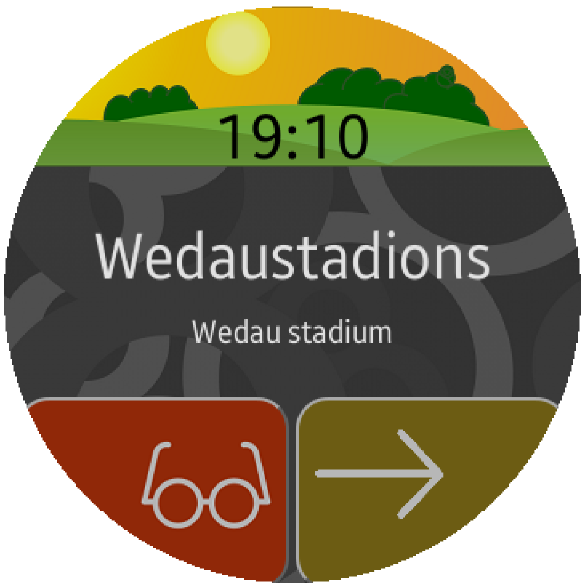

Figure 3.4: version 1

After the first discussion it was decided that the time should be displayed on 50% of the screen since the app would be use for checking the time and for learning words (see figure version 2.png). The time however was not readable enough and this was solved by changing the font color to white and by changing the background (see figure version 3.png). This new design worked really well and therefore this became the final design for the time.

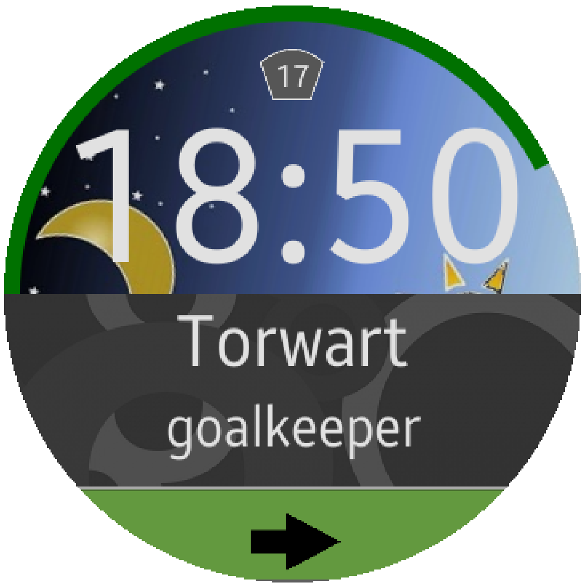

Figure 3.5: version 3

3.2.2 Words

The other important part of the app is showing the wordpairs, the word and the translation, the user wants to learn. Displaying wordpairs on a small circular screen was quite challenging which resulted in different designs for the watchface (see figure all possible versions.png). In the first design the wordpairs were placed in the middle of the screen (see figure version 1.png) with a white font.

Due to the circular shape there is more space available in the middle of the screen and therefore the wordpairs can be in a larger font. The color white was chosen to have a maximum contrast with the background since the background mostly consists of dark colors. The words have a larger font than the translations to have a clear distinction between the two types and later on the smaller font was selected to make sure that there is enough space to display the translations. Since the translations are placed lower than the words, less space is available. When only 50% of the screen was available for the wordpairs after the first discussion, the wordpairs were placed just below the middle where the space lost is minimal (see figure version 2.png).

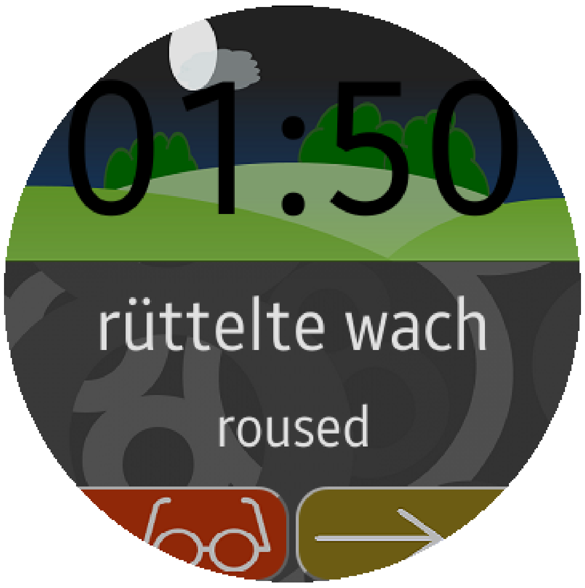

Figure 3.6: version 2

3.2.3 Background

The app consists of two parts, one for the time and one for the wordpairs. Both parts have different backgrounds. The background for the wordpairs was selected in the beginning of the project and did not change. The background consists of different dark grayish colors and with the white colored wordpairs the user should not have any problems reading the wordpairs.

At first the idea for the background of the time was that it should support the current time. Therefore four different backgrounds were designed for the morning, afternoon, evening and night (see figure first designs background.png).

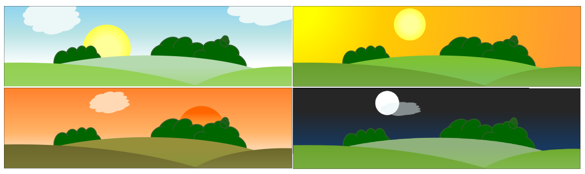

Figure 3.7: first designs background

The space for the time then changed to 50% of the screen and therefore the backgrounds were unusable because of the old dimensions of the images. Instead of changing the images to the new dimensions this moment was used to think again about the background of the time. The different images for the different types of the day had to be saved on the watch. This storage issue was solved by using one image. The new background consisted of the sun, the moon and a transition between blue and black. This background rotated around it's center which makes it possible to support the current time with the positions of the sun and the moon (see figure day_cycle_2.png).

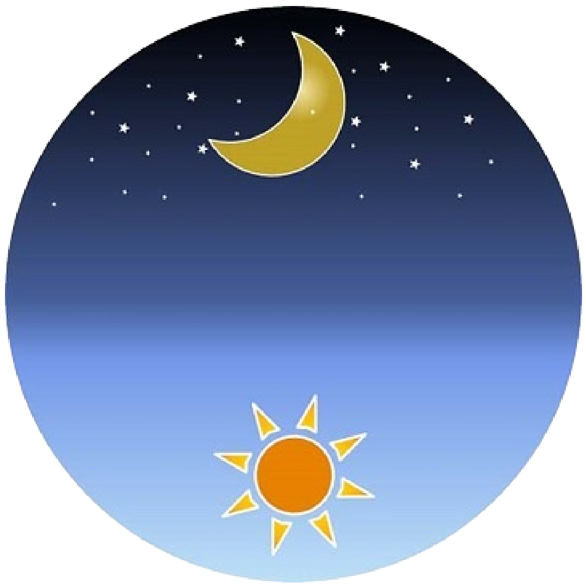

Figure 3.8: day cycle

Due to the rotation a sunset and sunrise could be seen on the watch. To emphasize the sunset and sunrise a landscape was added to make, for example, the sun appear from the horizon with a sunset. The used colors for the landscape were different from the colors in the rotating background and hence a new background with the sun and moon was designed to match the landscape (see figure version 5.png). The landscape worked really well with the rotating background and therefore two other landscapes were designed from which the user can choose (see figure designs landscape.png).

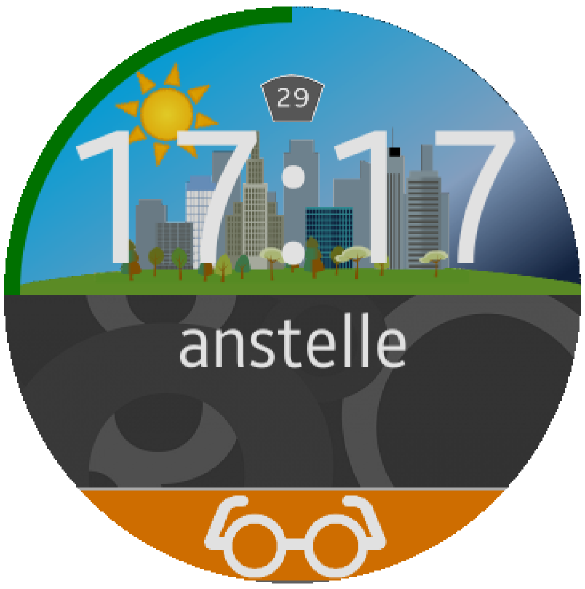

Figure 3.9: version 5

Figure 3.10: landscapes

3.2.4 Buttons

As mentioned before swipe detection was insufficient so it was decided that the app should only use touch events. To navigate through the app with only touch events it seemed logical to use buttons. With the time on top of the screen and the wordpairs in the middle there was unused space in the bottom for the buttons. In the first design the user should be able to reveal the translation of the shown word or to go to the next wordpair. This resulted in two buttons, a red one with glasses on it and one with an arrow to the right (see figure version 1.png). Glasses were chosen for revealing because it could refer to looking something up and that matches with revealing the translation. Since the app is displayed on a small screen, the buttons should not be to small and thus almost all the available space in the bottom was used for the buttons to ensure that there was enough space for the user to press a button. Different tests showed that the buttons could be made smaller. This was a favorable conclusion since after testing it was decided that only 50%

will be used for the wordpairs (see figure version 2.png). The design of the buttons was not really pleasing and needed some rethinking. The first modification was erasing the reveal button. Instead of the button the user could touch the word to reveal the translation. The 'next' button was widened so it covered the bottom of the screen (see figure version 3.png). During development more options were implemented and therefore a new button was made next to the 'next' button. The settings button gave access to these options (see figure version 4.png). The curves on the buttons as can be seen in the first design were replaced with right angles to create a modern feeling. The app was tested and more attention was paid to improve the order in which the wordpairs are shown. This process should become smarter by use a knowledge estimator that would estimate what the next word should be to maximize the learning process. This estimation could only be made when the server could receive feedback of every single word. It is therefore necessary that the user should give feedback after every word. The design of the app was adapted to this new idea by changing the 'next' button to a 'reveal' button (see figure version 5.png). When the user pressed this button the translation appeared and the 'reveal' button changed in three different buttons: a 'wrong' button, a 'menu' button and a 'right' button (see figure version 5 reveal.png). This new design forced the user to give feedback after every word. This feedback could be used to optimize the order of the words. The first design of the buttons were a cross and a checkmark to let the user know what to press when a word was not know or was known. Later on a book and a graduation cap were used for the same purpose but the feedback to the user would be less harsh with the new design (see figure version 6 reveal.png).

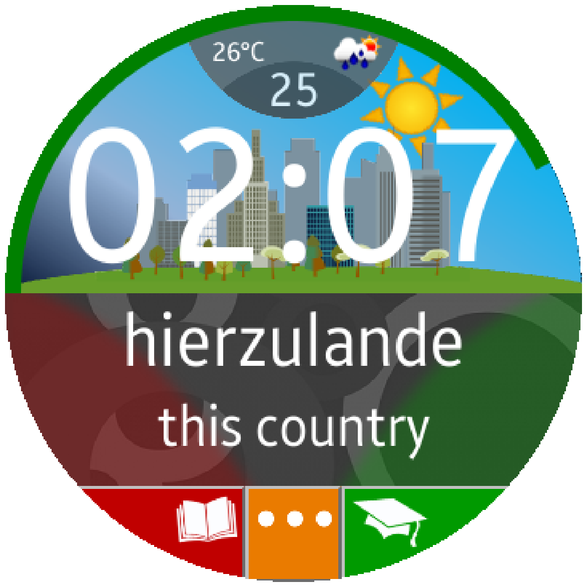

Figure 3.11: version 6 on revealing

3.2.5 Information components

Besides the time the app also shows the date. The date was added later when the design of the time was completed. In the first place the date had his own small icon on top of the screen (see figure version 3.png). This icon was partly covered by a Tizen icon for the swipe down menu. Therefore the icon for the date was moved downwards near the time (see figure version 5.png). During a discussion it was mentioned that it would be nice to be able to see the temperature and the weather type on the watchface. After some research a free api* was found that could provide the app with this information. In the beginning of the implementation the temperature was placed to the left of the date and the weather type was placed to the right of the date. This was barely readable and it did not fit in the realized design hence a new icon was designed that contained the date, temperature, weather type and an open area in the middle for the Tizen icon for the swipe down menu (see figure version 6.png).

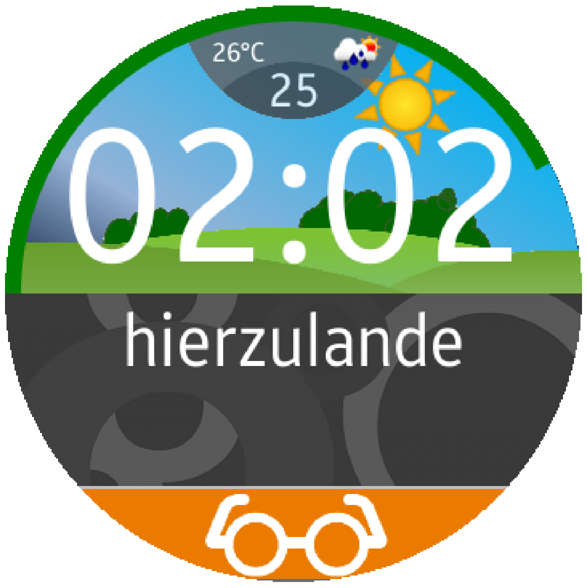

Figure 3.12: version 6

3.2.6 Option lists

The app has access to two different option lists: the settings which can be reached by double tapping on the time and the menu that can be reached by tapping in the middle of the translation. This was different in one of the first designs. When the app was able to show the words with the translation, the user should have an option to tell the app that a word had a wrong translation or that a word was learned and thus in both cases it should not reappear again. Due to the small screen it was not convenient to add more buttons for each of these options to the watchface and therefore the idea came to make a menu with these extra options. Besides these options the user should be able to reverse the order in which the wordpairs were asked (when the words were displayed from German to English, the words would be displayed from English to German after the button was pressed), to insert the number of words that the user wanted to learn and to log out of the app. All these options were reachable for the user by a ’menu’ button in the bottom. The ’next’ button was replaced by two buttons, the ’menu’ and the ’next’ button (see figure version 4.png).

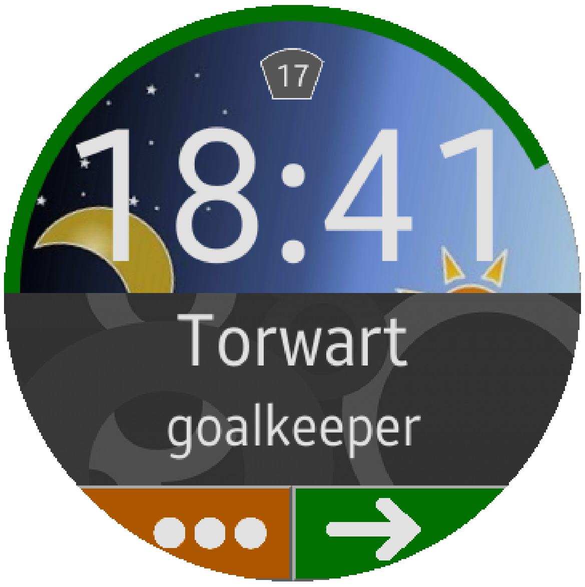

Figure 3.13: version 4

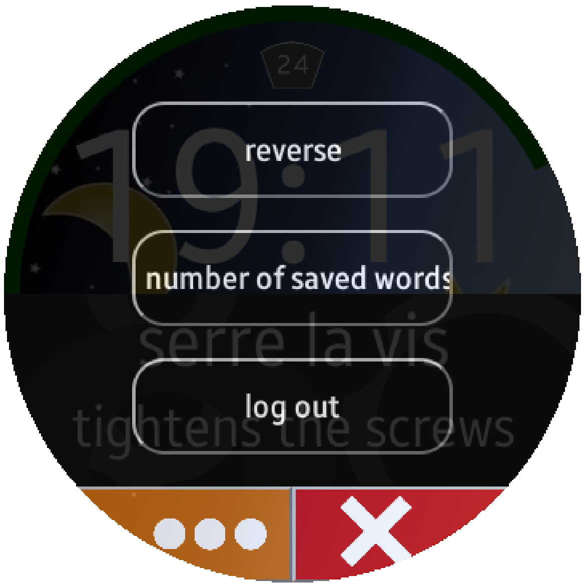

Figure 3.14: version 4 settings

When the 'menu' button was clicked a menu would appear with the 'trash' button and the 'I learned it' button. In the bottom the user could close the app or open the settings (see figure version 4 menu 1.png). By pressing the 'settings' button the user would open the settings page with the buttons for reverse, number of words and log out (see figure version 4 settings 1.png). After some discussions the overall opinion was that the 'menu' button should not be this large on the watchface since the menu would give complementary options which were not part of the main function of the app. Therefore the button should not cover half of the bottom and a solution was to split the menu and the settings to two different pages that the user could reach independently. The user could reach the menu by tapping in the middle of the translation (see figure version 5 reveal.png) and the settings could be reach by double tapping on the time since this space was hardly used. The double tap was necessary because tapping the time would change the background. The menu and the settings both had a black transparent background for a long time. This was inspired from the transparent and blurred swipe down menu from a well-known OS. Unfortunately the blur effect was not a success, but the transparency remained.

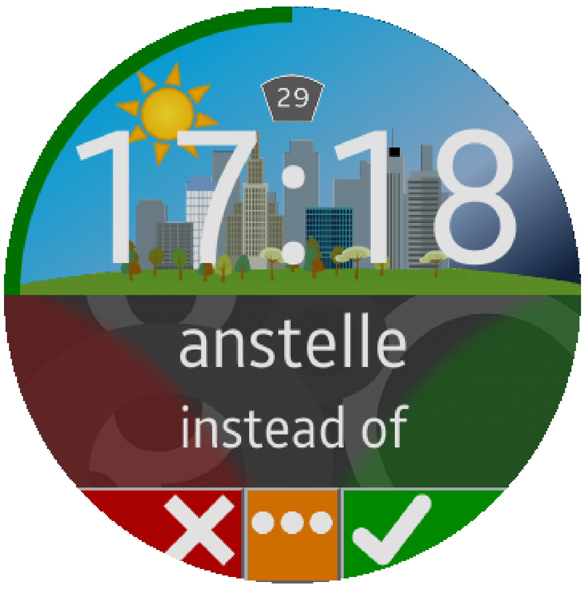

Figure 3.15: version 5 reveal

During testing the text in the buttons were hard to read and the buttons of the main page were visible through the buttons on the bottom. Therefore it was decided that the menu and the settings page should have a background. For the background the same image was used as the background of the wordpairs to preserver the unity of the design. A lot of translations depends on the context of the word. Hence the user should be able to see the context of the word and this option was added to the menu page (see figure version 6 menu.png).

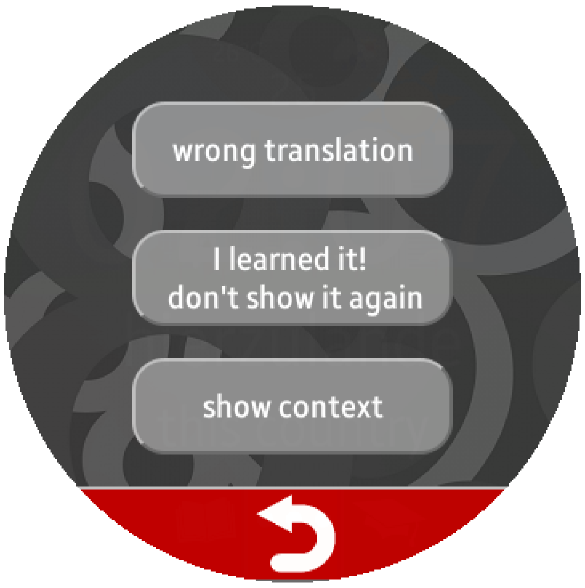

Figure 3.16: version 6 menu

Later on the idea was abandoned of letting the user choose how many words the user wants to learn and instead came the ’profile’ button as the latest addition to the settings page (see figure version 6 settings.png).

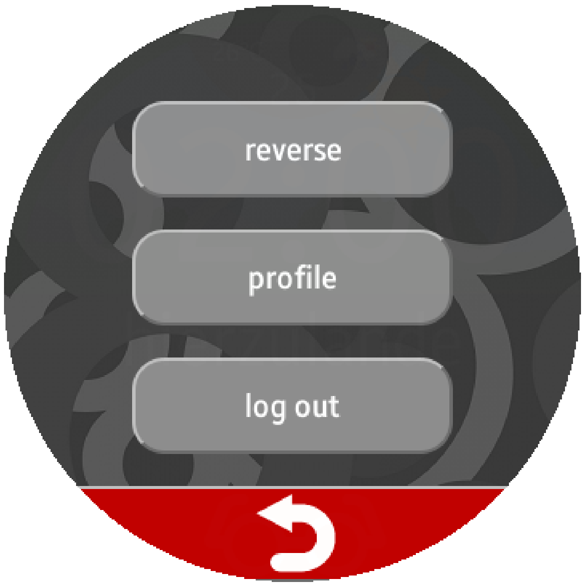

Figure 3.17: version 6 settings

3.2.7 Profile

One of the latest addition to the app was the implementation of a profile page where the user could see the four medals that could be earned by using the app (see figure version 6 profile.png). The four categories are: words learned, total time, longest session and longest streak. This page was inspired from a presentation where gamification was mentioned. The idea was that when users could earn medals, users would be more willing to use the app. When the user presses the 'I learned it' button, the number of 'words learned' will increase and after 10 new learned words, a popup will appear with a motivating message to continue. The total time the user used the app is also registered. When this time is the same as one of the pre-determined minutes a popup appears (see figure version 6 popup.png). With longest session the time is measured that the user uses the app continuously and with the longest streak the number of days is registered in which the app is used daily. One day of not using the app resets the longest streak.

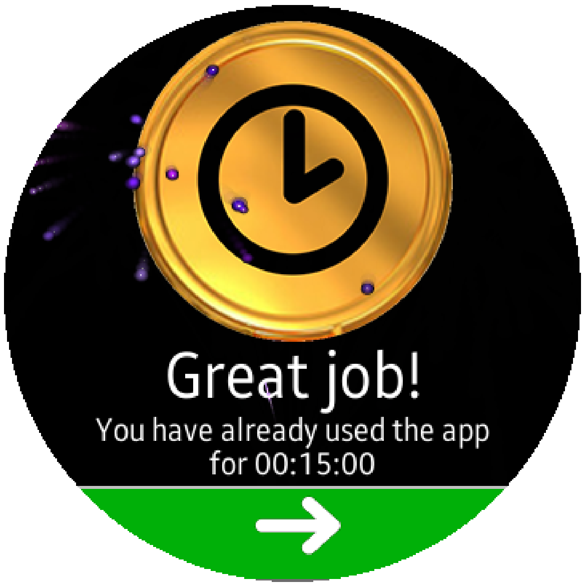

Figure 3.18: version 6 popup

3.2.8 Effects

To improve the experience of the app some effects were added: to give feedback after a button was pressed, to give information (why an option was not available or why a medal was earned) or to beautify a popup. Because of the small screen the user could think the wrong button was pressed and therefore the user should get feedback about which button was pressed. The most important buttons in the app are the buttons used for indicating whether a word was known or not. When one of these buttons is pressed a green or red image appears depending on if right or wrong is pressed. This image stays on the display for a few milliseconds before it fades out. In the app several popups are added to give the user feedback when an option is not available. The design of the popup is based on the first design of the option lists. The popup has a black transparent background with white text on it. The popup could appear when: there is no connection with the internet, when a wrong code is inserted, when the user has too few words, when there are too few words left on the watch or when a user earned a new medal. The popup that appears when a new medal is earned is beatified by some fireworks.

4
The Implementation

In this part the implementation choices are described. In this project writing the code was one of the biggest challenges. The code has been changed a lot during the process. One of the reasons was to increase readability, but also because of design changes and new features which had to be implemented. The code is written in JavaScript in combination with HTML 5, this means it is web-based application. The code is build with a framework called requirejs, this makes it possible to have multiple files (modules) in javascript, this to increase readability and structure. In the following sections, we will zoom in some interesting and important decisions and choices related to the implementation.

4.1 Flash card algorithm

For presenting the words we used a certain algorithm. This algorithm will make sure users will learn faster and it's based on the flashcard method. The algorithm works in such a way that words which are answered wrongly will be faster repeated than words which were answered correctly. As explained in the design the user can indicate whether he had the answer wrong or right in his head. If the user had it wrong the word will be repeated after five more words, when the user had it right in his head the word will be repeated depending on the number of times the user had it correct. The word will be $timesCorrect * 5$ positions moved in case the user the had it right in his head. In figure 4.1 this process is shown.

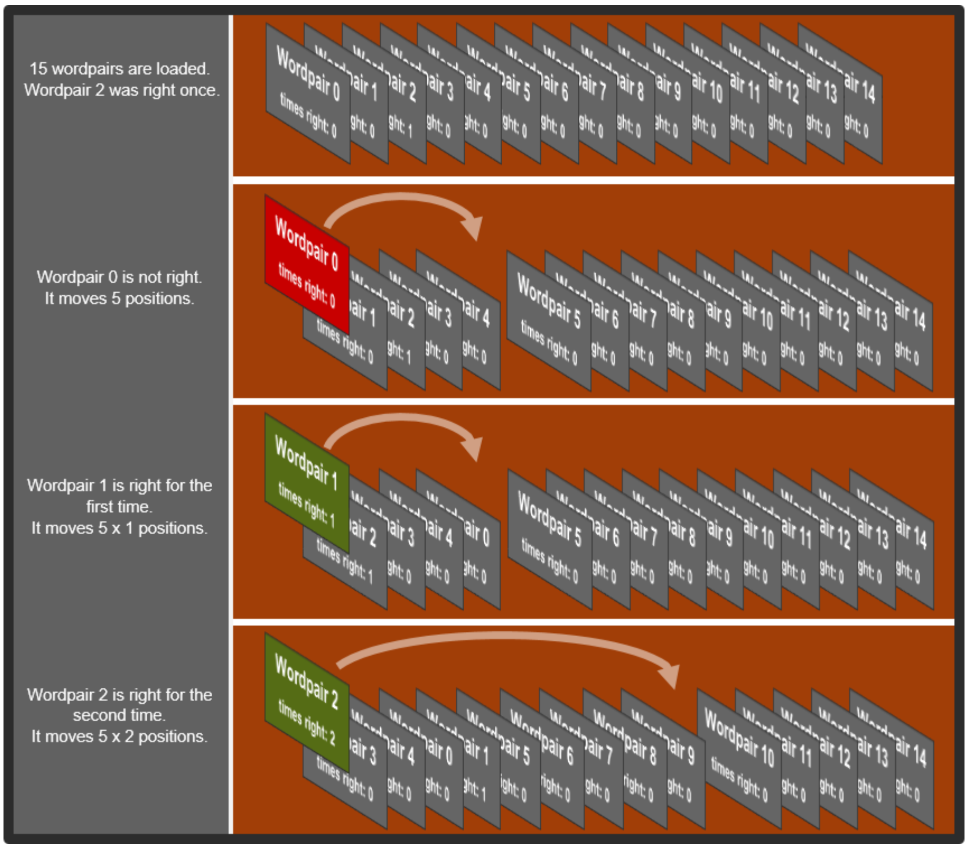

Figure 4.1: Flashcard simulation

In listing 1 the code involved is shown, every time the users taps wrong or right the function $updateWordPair$ is called, if the user had it right the number of times correct will be increased. After that the current word will be moved in the array. This happens by first adding the word some positions further and then deleting the current word (i.e., deleting element zero in the array). If the user taps wrong the current progression will be reset (i.e, timesCorrect becomes zero again), and the word will be moved five positions to the right (see figure 4.1). This means the word will be repeated reasonably fast. Because the first element of the array is always shown on the screen, the next word will be drawn on the screen, since the previous first element is deleted. In $userData.js$ everything related to the user is implemented, so this is the place where words can be found. A wordPair is characterized with the following attributes: word, translation, id, context and number of times correct. Getting the word pairs from the server is explained in the next section.

```
    updateWordPair: function(wordIsRight) {
      if (wordIsRight) {
        wordPair[0].timesCorrect++;
        wordPair.splice(wordPair[0].timesCorrect *
   NUMBER_OF_FLASHCARDS, 0, wordPair[0]);
      } else {
        wordPair[0].timesCorrect = 0;
        wordPair.splice(NUMBER_OF_FLASHCARDS, 0, wordPair[0]);
      }
      wordPair.splice(0, 1);
    },
```

Listing 1: flashcard implemenation in userData.js

4.2 Getting new words from the server

In the module session.js the words are fetched from the server. This session is created in the main.js. The code which login gives (either entered by the user or loaded from the userData) will be used in session.js to get the words for the user. The session can get new words with this unique code. The code is generated once a new account is created and can be found in the account when logged in the browser. The code contains eight digits. When a session is created, the first thing to look at are there currently any words on the watch. If there any words on the watch we can set the status to ‘succes’, in case there are no words on the watch yet: the watch has to communicate with the server and get the words with the endpoint ‘bookmarks_to_study’. This endpoint will return the words which are currently the most important to study for that particular user. In the implementation we made the choice to get fifty words in case the user has no internet for longer periods of time. The session can return different states: if the words are successfully fetched from the server the state will be ‘succes’ as described earlier, if the code is invalid the state will be ’wrong session number’, in case of no internet connection the state will be ’no connection’. There is one exception, but this will not happen very often this is when there are too few words in the account of the user. One of these states will be returned to the main module and the main module will give this state through to the login module, in the login the user can then be informed by a popup that will inform the user about a specific situation. If the state was ‘succes’ the user will get in the main screen where the words are presented. The login module is not the only place where the watch tries to get new words, it also happens when the user is already logged in and a screen on event occurred (i.e., user makes arm twist to look at time). The new words will then be added to the current wordlist on a screen off event (i.e., user makes again a arm twist to indicate he is not looking anymore), adding words at this point is to prevent any conflicts: adding words to a list which the user is currently doing stuff with. New words will only be added if the list is smaller than fifty words. The list can become smaller if the user has marked a couple of words as 'learned it' or as 'wrong translation'. When writing the code the initial preference was to get words and add them at the same time on a screen off event, since the user won't be using the application at that moment and it would really feel as everything happens in the background. Unfortunately this didn't work the watch gets in a sort of sleep modus and won't be able to communicate with server anymore. The GET request wasn't executed so it was impossible to do this on a screen off event. When the watch tries to get new words from the server, this happens asynchronously to prevent the watch from getting slow. Implementing this asynchronously was huge in the eind, because the user is able to get a lot of words in his account (e.g., some user had already 10000 words) the endpoint will be really slow, since the list of words is not already sorted on importance on the server server side. In the first implementation when it was implemented synchronously, therefore the user could have some delay before being able to interact with the watch. It was first implemented in this way to prevent conflicts, user cannot do something with a list while words are being added. This is now solved by getting the new wordPairs on a screen on (see listing 2) and add them later on a screen off (see listing 3).

```
function getNewWordPairs(newWords, currentWords) {
  if (currentWords.length === 0) {
    return newWords;
  } else {
    for (var j=0; j<currentWords.length; j++) {
      for (var i=0; i<newWords.length; i++) {
        if (currentWords[j].id === newWords[i].id) {
          newWords.splice(i, 1);
          break;
        }
      }
    }
  }
  return newWords;
}
```

Listing 2: getting new wordPairs in session.js

```
  addWords: function(numberOfWords, newWords) {
    wordPair = wordPair.concat(newWords);
    wordPair = wordPair.slice(0, numberOfWords);
  },
```

Listing 3: adding new words to wordPair in userData.js

4.3 Usage Tracking

4.3.1 Events

Events are implemented to give the knowledge estimator information which can be used to analyse the knowledge of the user. The knowledge estimator is on the server side implemented. This means we had to decide at what point we should send the events. Because the knowledge estimator decided what the next word is going to be for the user, the events should be send when the user taps wrong or right. This is the moment the user will be presented with a new word. The knowledge estimator can receive the following events: reveal, right, wrong, wrong translation, I learned it, showContext, screenOn and screenOff.

4.3.2 Clicktracking

The clicktracker is designed for research purposes. It's purpose is to track where the user clicks/taps, so the coordinates are saved. Not only the click position is saved, also the click type (e.g., user taps on reveal). For some function this can be really interesting like the right button; the user can tap on the button itself or on the green space above the button. Both actions result in the same thing. In the future we could change the design based on these results to give the user a better experience. To conclude the clicktracker is nothing functional, it is purely for research purposes.

4.4 Changing the watchface and sending and saving

Sending and saving the clicks and events was a big challenge in this project. This is the result of the user being able to change the watch face at every moment, which will result in losing data. Because the user is able to go a different watch face at any moment clicks and events should be saved to the storage of the watch immediately. We do this by saving it to the local storage. This is is implemented in the same way as saving to the local storage in a web browser. The application is web-based using JavaScript and HTML, which was already explained earlier on. A click and event are immediately pushed to the storage after they occurred. The clicks in the storage are send on a screen on event asynchronously, so it won't influence the speed of the application and it happens in the background. When the clicks are successfully send to the server, the server will return 'OK', if the watch then receives this message the clicks in the storage can be freed, they are now at the server. The events in the storage are send on 'right' and 'wrong' as earlier on explained to inform the knowledge estimator. Just as with the clicks the events are deleted from the storage if they successfully have been arrived on the server side. Losing data with changing the watch face not only happens with clicks and events, it also happens when the user reverses the way the words are asked (e.g., English - Dutch to Dutch - English) and with changing the background. The login code is being saved the first time and used for automatic login after a watch face change.

4.5 Drawing the userinterface

The userinterface is drawn every second on the screen, this happens in gui.js in the draw function which is public (shown in listing 4). The main module has an function which calls this function in gui every second (shown in listing 5). The screen is thus updated every second. Time and profile are also being refreshed in the draw function, in the future it would be nice to have these in a different function to make sure draw only draws stuff.

```
  draw: function() {
    profile.refresh();
    time.refresh();
    time.draw();
    time.drawDate();
    battery.draw();
    weather.draw();

    var totalMinutes = time.getHours()*60 + time.getMinutes()*1;
    background.rotate(weather.getSunrise(), weather.getSunset(),
 totalMinutes);
  },
```

Listing 4: draw function in gui.js

```
function updateScreenEverySecond() {
  gui.draw();
  setTimeout(updateScreenEverySecond, 1000);
}
```

Listing 5: main class

5 The Evaluation

describe how you evaluated to show that your approach was successful. You may need a methods section, a results section and a conclusion section.

6 Conclusion and Future Work

summarize your thesis again as in the introduction. Describe how your evaluation revealed that your system is successful. Describe future work in this area.