ISSN: 2664-9640

Design and implementation of professional English reading website based on Android platform

Jun Wu

School of Arts and Sciences, Yangtze University, Jingzhou, China

Abstract

This article uses several stages of work such as demand analysis, algorithm research, system design and coding, and focuses on the statistics and analysis of new English words, the intelligent collection of reading materials, and the application of Ebbinghaus memory curve theory in English reading. The core issues are algorithm design, system architecture design, etc., the Ruby on Rails development framework is used to develop the Web system, and the mobile Internet and smart phone technology are used to improve the usability of the Web system and improve the reader's professional English reading efficiency.

Keywords

English reading, network technology, mobile Internet, Android, Ruby on Rails.

1. Introduction

At present, there are many websites that provide word-recitation services in China, such as scallop.com and lovewords.com, but the main mode is to memorize words. With the help of phrases and example sentences to assist understanding and memory, users need to use a fixed time to do it. Because the subjective will is not strong and the objective conditions are limited, fewer people can insist on using this model. Moreover, such services will have a certain impact on the pace of people's lives. The research background of this project is inseparable from the basic conditions of the English learning information environment. From Kingsoft PowerWord to Baidu translation, from Jinshanbei word software to scallop, due to the limitations of technology and network environment, it lasted ten years.For the rest of the year, the environment and methods of English learning are mainly based on words as the core object to improve vocabulary.The goal is to use simpler word memory as a means.

2. System architecture and technical route

2.1. Choice of technical route

The system architecture of this project is further clarified on the basis of the determination of the technical route, and is constructed gradually by fully following the design philosophy of Ruby on Rails, which is different from the general design ideas. The main reason for choosing Ruby on Rails as the development platform is that it can meet the technical requirements of software development mentioned in the previous article, and the obligatory object-oriented features of the Ruby language and its flexible, easy-to-read and elegant design style can greatly reduce project development. And maintenance costs, and widely used languages such as Java, PHP, C#, phtyon have certain deficiencies for the above needs.

2.2. Project development environment

The establishment of the Ruby on Rails development environment is a relatively complicated issue, which requires the participation of many elements such as the operating system, ruby language, rails framework, compatible class libraries and component programs, auxiliary tools,

ISSN: 2664-9640

etc. to proceed smoothly. The Ruby on Rails development environment has the best compatibility in the MAC OS system, but due to its hardware limitations and higher capital costs, it was chosen to be installed and deployed in the Ubuntu 13.04 release operating system, and the developed web The operating environment of the system is Ubuntu Server 12.04

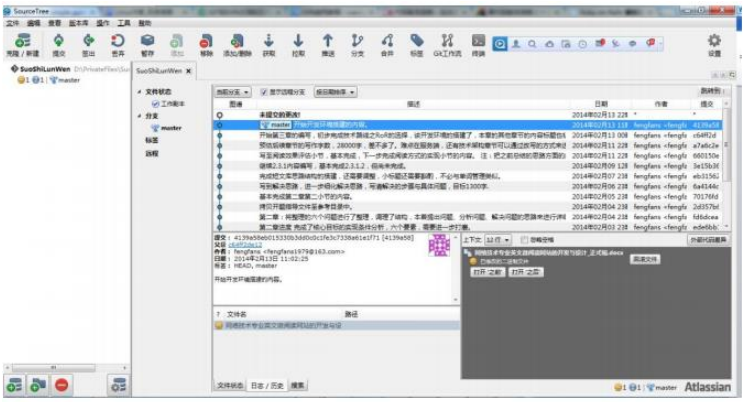

Figure1.The operation interface of Source Tree

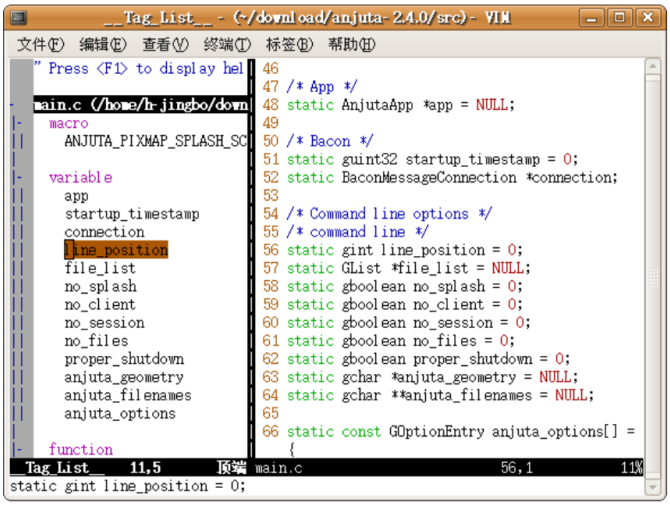

Figure2.VIM code editing interface

Figure3.Rails development interface for TextMate 2

2.3. Selection of development aids

During the development of this project, we are using Source Tree, an open source git graphical interface client software, to manage code warehouses. This software supports Chinese and mainstream code warehouse management websites. It is also relatively easy to use. It is suitable

ISSN: 2664-9640

for windows, linux and The good compatibility of MAC OS system is widely welcomed by programmers. At present, the preparation of this article is also using it to cooperate with git.

2.4. Mobile terminal platform

To develop client applications running on smart phones, the choice of target operating platform is mainly to consider the market share of IOS and Android systems, especially the target group of this project-the actual situation of college students at school to measure this project Android is preferred as the target operating platform for the first mobile terminal application to be developed and supported.

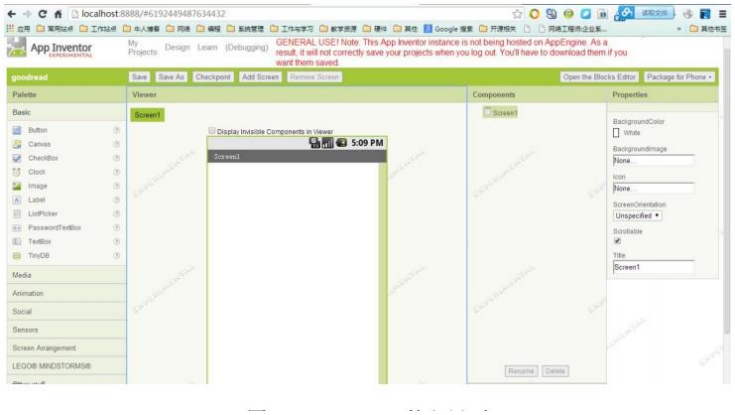

Figure 4.Appliventor's main interface

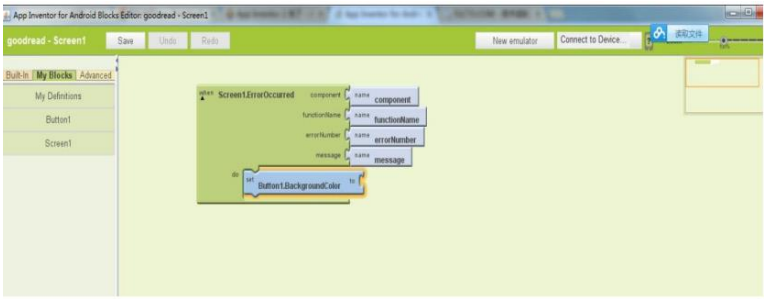

Figure 5.BlocksEditor editor interface

3. Core algorithm and function design

3.1. Core issues

Since the design goal of this project is not a more general application project, it has a more mature solution. To achieve the established design goal of this project, some core problems need to be solved. Around these core problems, effective algorithms need to be designed to achieve it. It has a fixed function and must be fully integrated with other functional modules. This link is the key to achieving the predetermined goal. Combining the explanations in the previous chapters, several more complex issues can be sorted out from the established requirements. These issues are summarized as follows:

3.1.1 Analysis of new words in short essays [1-2]

The problem of new words analysis of short essays is the first core problem that this project needs to solve. This problem specifically solves the new words and number of readers in the analyzed essay, and performs necessary analysis on the basis of this basic data, and lists Statistics are used as a basis for decision-making on whether the reader chooses to continue reading the essay.

3.1.2 The memory problem of new words [3-4]

ISSN: 2664-9640

The generation of new words comes from the comparison process between the comparative essay and the vocabulary list in the core question 1, and is selected from the analyzed essay. The memory here refers to the memory problem of the reader's brain, not the memory problem of the computer. The main content of the core problem to be solved is to analyze and sort several new words generated by the program module generated according to the Ebbinghaus memory curve in the process of comparing the short essay in the core problem 1 with the word list. , And with the original vocabulary list (the words memorized by the reader) in a certain periodicity and regularization, it is combined in batches and sequentially and cyclically, as a new vocabulary list for the reader, to compare with the short text, thus As a necessary link to promote readers to absorb and digest new words and finally convert them into memorized words.

3.1.3 The problem of intelligent collection of short essays

The core purpose of solving this problem is to provide readers with richer reading materials, and the purpose of these materials is to solve the following two requirements.

The first aspect is to serve the needs of searching for reading materials with a higher matching degree for specific new words in the memory problem of new words. The second aspect of the demand is the demand for collecting reading data based on the analysis of the reader's basic data.

3.2. Algorithm design and research results

3.2.1 Analysis of new words in short essays [5-6]

The problem of analyzing new words in short texts is the simplest of the above three problems, and the program execution environment involved is also the simplest. On the basis of satisfying the problems of vocabulary list, short text storage and extraction, only the comparison of the two is needed to complete the whole process. You need to extract the data from the database and save it to a variable, and convert the word name to an array variable before comparing. In the regular comparison process, a series of formatting operations are also required for the short text, including operations such as removing duplicate words, deleting "`s" characters, etc., converting it into a word array and comparing it with a word list array, extracting the birth word array and storing it in a variable , And display relevant statistical information in the interface.

3.2.2 Research results on the memory problem of new words

As mentioned earlier, this problem is a relatively complicated one. The complexity of this problem is that the problem in this project is not simply to solve the process of sorting new words in batches and simply memorizing words, but to order each batch of words. The vocabulary list is used as a keyword to search for suitable short texts and push them to readers for the process of reading.

After the problem is decomposed, it needs to be solved through the following steps:

(1) The design follows the basic algorithm of the Ebbinghaus memory curve, and writes the ruby version of the code.

(2) Analyze how to design the memory curve time point more reasonably. In this link, the code in step (1) should be used to participate in the experiment and test.

(3) Re-organize the system architecture that has been implemented, consider how to adjust the architecture design to meet the realization of this function, and ensure that the numerical changes at the time points of the memory curve in the future should be more flexible, and there will be no changes in the system architecture or a large amount of code. modify.

(4) Carry out specific coding practices to realize the designed system functions.

3.2.3 The problem of intelligent collection of short essays

ISSN: 2664-9640

The intelligent collection of short essays is a service that runs regularly in the background of the system. It collects information through a combined search of keywords. As explained in the analysis chapter of the previous problem, this is a problem that requires continuous research, analysis, and adjustment. There is no clear basis for judging its good or bad standards. Therefore, this article can only discuss the specific ideas and solutions to the problem. The process is described.

The overall solution is to first collect short text materials of specific subjects by search engines and save them in the system short library; then use the keyword groups or keyword groups formed by the author's basic data to compare the short library for a second time. So as to push it to readers.

3.3. Functional design

Explains the solution ideas of the core problems and the description of the research results. Next, the functional design of the system will be described in detail to provide a follow-up basis for subsequent code writing.

3.3.1 User Management Module

The user management module is the foundation and center of the entire system, and other functional modules will be implemented around it. Taking into account the future functional requirements, the functional design of this module should meet the following requirements.

(1) Users can self-service user registration, login, modify registration information and password, reset password, etc.

(2) The user can register with a character string or mailbox as the user name.

(3) In the future, the account can be mutually recognized with Sina Weibo, Tencent QQ, etc., and the accounts of these systems can be used for registration and login in this system.

(4) Ensure that the user password is encrypted and stored in the database management system together with other registration information.

(5) Ensure that different permissions can be set flexibly for different types of users in the future.

3.3.2 Word List Management Module

The vocabulary list management mainly solves the maintenance and management function of the reader's private vocabulary list, which has a one-to-one relationship with the reader user. The specific functional design content is as follows:

(1) Each reader has a vocabulary list, and other readers cannot view and modify it without authorization.

(2) Each word list saves word names and remarks information.

(3) Readers can self-maintain the word list, add, delete, and modify words. Next, the author will develop the word query function.

(4) The word list of the reader cannot be deleted.

3.3.3 Short article management module

The short article management module mainly solves the management problem of the reader's private short library. It has a one-to-one relationship with the reader and the user. The specific functional design content is as follows:

(1) Each reader has a short library, and other readers cannot view and modify without authorization.

(2) Each essay includes data such as title, keywords and body content.

(3) Readers can maintain a library of short texts by themselves, add, delete, and modify short texts. In the next step, the author will develop a short text query function.

(4) The short library of the reader cannot be deleted.

ISSN: 2664-9640

(5) In the essay list, the link of each essay record contains a comparator function, which can be compared with the reader's word list and display statistical information, including the number of new words, a list of new word names, etc.

3.3.4 Short essay and word list comparator module

In addition to the realization of the comparator function in the essay management module, a comparator module will be developed for readers to compare with the vocabulary list when selecting essays to assist readers in self-organizing and collecting essays. This functional module can be accessed by every reader after logging in. After entering the content, the comparison can be performed and the same statistical information as above will be displayed on the new page.

4. Web system coding and testing

4.1. The operating environment of the Web system

The operating environment of the Web system in this project is a relatively complex operating environment, which includes operating systems, tool software or programs, language interpreters, development frameworks, and class libraries. The operating system selection is naturally the aforementioned Ubuntu, but in the project The version used in the development phase is different from the version used in the deployment phase.

The operation of Ruby on Rails also requires some third-party tool software or libraries, which mainly solve the dependency problem of ruby class library and rails gem operation, as well as the dependency of development tools and auxiliary software (such as git and vim). The list of these procedures is as follows:

Wget vim build-essential openssl libreadline6 libreadline6-dev libmysqlclient-dev curl git-core zlib1g zlib1g-dev libssl-dev libyaml-dev libxml2-dev libxslt-dev libcurl4-openssl-dev autoconf automake libtool imagemagick libmagickwand-dev libpcre3-dev nodejs libpq-dev

4.2. The development process of functional modules

The user management module of this system not only solves the problems related to user registration and login, but also meets the supporting role of some important functions of the system in the future. In summary, there are two: One is to ensure the configuration of more complex user permissions The second is the support for user account authentication of major mainstream systems. In the Rails community, there are more mature gems or combinations that support these two needs. Combining the suggestions of reference materials, I chose devise. This gem can be integrated with omniauth[26] (gem package that specifically supports third-party verification) to meet the needs of third-party authentication. It can be integrated with cancan to expand the requirements for permission settings and provide reliable guarantees for the project to go online as soon as possible. Effective support.

Figure 6.Site login page

ISSN: 2664-9640

Figure 7.Website registration page

4.3. Word List Management Module

The vocabulary management module is the basic module to realize all the functions of this project. It is mainly used to realize the function of self-management of private vocabularies by users. It has a one-to-one relationship with the user model, and it has a one-to-one relationship with short text management. . The function realization of this module does not depend on a specific external system environment, and the rails core gem can realize the relevant functions. Similar to the user management module, it needs a back-end database to provide support for the data storage of the word list. The other functions of this module are implemented by ruby in the rails framework. The function of this module requires two models to implement specifically, one is the wordlist (wordlist) Model, a wordlineitem (word list item) model.

Figure 8.word list query page

Figure 9.Show newly added words

4.4. Short article management module

The short essay management module is a relatively simple and independent module compared to the word management module. Its main task is to solve the problem of adding, deleting, modifying and checking short essays, solving the problem of comparing short essays and

ISSN: 2664-9640

vocabulary lists, and is specifically responsible for solving new word analysis The functional module of the problem.

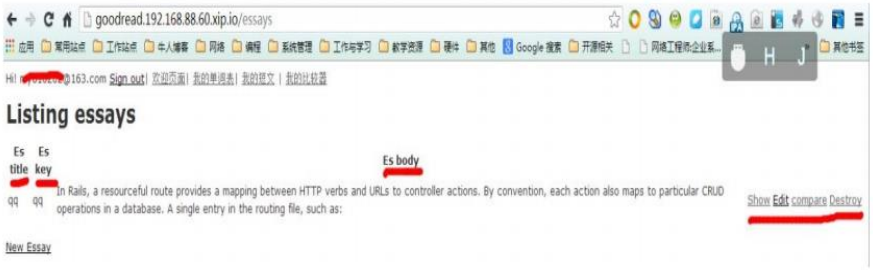

Figure 10.Short essay list view

4.5. Android Client Module

Finally, it is the development content of the android client of this system. Due to the tight time and limited development experience, the development of this client module failed to achieve the goal of native android applications to access system data, but the system's homepage address was packaged into The app runs on smart phones or other types of mobile terminals, and currently does not support the IOS platform.

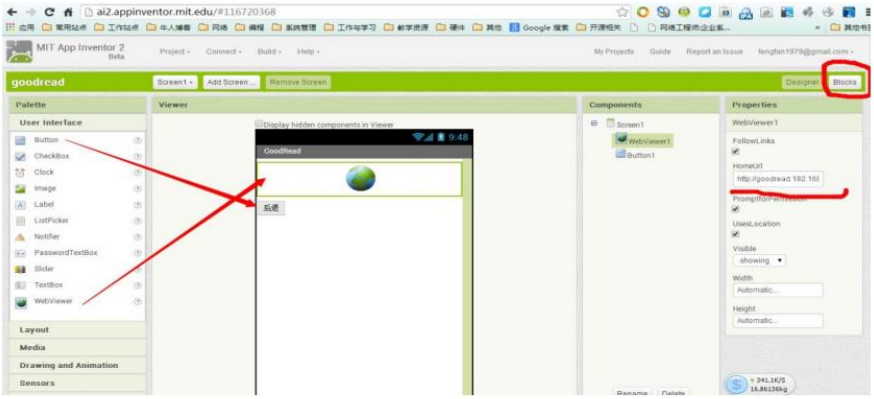

Figure11.Appliventor interface drawing window

Figure12.Smart phone APP running interface

Drag the Button control from the control list on the left to the bottom of the WebView control, change the button title to Back, click the button delineated by the red circle to open the code editing (Blocks) window, you can start the code like playing a drawing game Preparation. Next, click the button circled in the upper right corner of Figure 5-16 to open the interface in Figure

ISSN: 2664-9640

5-17, and complete the following two function codes: one is the behavior of opening the WebView home page by setting the window initialization event to load the system home page; It is to set the WebView control back behavior for the Click event of the button to perform the page back operation when the button is clicked.

Then, go back and click the connect——>AI Companion command at the top of the window to open the running interface of the phone.

At this point, the application code writing and debugging work is completed, the last step is to package this application and install it on the smartphone, the packaging method is Build-app (provide to computer for .apk)

References

[1] Qi Jingjing. Ebbinghaus Memory Method and English Learning. Journal of Suzhou Education College 2011,01.

[2] Feng Ling et al. Application of Ebbinghaus forgetting curve in word memory. Journal of Chengdu Aeronautical Vocational and Technical College, 2007,09.

[3] Jiang Fengxia, Guan Lingyong. How to use the law of memory forgetting to improve English memory. Journal of Sun Yat-sen University, 2007, 08.

[4] Peng Huiling. Research on the Promoting Effect of the Mini-Pass Memory Method on the Comprehensive English Proficiency of Learners of Different Levels. Journal of Jinggangshan University, 2009,07.

[5] (Japan) Takahashi Seiyoshi, Goto Yuzo. Ruby Programming--Learn programming from the father of Ruby (Boshuo Culture). Electronics Industry Press, 2009.

[6] (Japanese) Aoki Minero, Goto Yuzo, Takahashi Seiyoshi. Ruby programming 268 techniques (2nd edition). Publishing House of Electronics Industry, 2009.