УДК: 004.420

“Pernetakta” system for fast learning to typing

Massalimova G
Suleyman Demirel University

Computer today is very important for everyone. But this is not just a matter for different games or the like is a means for fulfillment of various goals. Computer today helps to do what used to do a few people and a lot of time.

Speed the task of the computer may depend not only on the obvious factors: the performance of the processor, or, roughly speaking, human ingenuity. Sometimes the best deterrent is purely mechanical work, exactly typing on keyboard[1]. Does it matter how we type? Yes. Touch typing allows us to write without thinking about how we are writing, freeing us to focus on what we are writing, on our ideas. Touch typing is an example of cognitive automaticity, the ability to do things without conscious attention or awareness. Automaticity takes a burden off our working memory, allowing us more space for higher-order thinking. The keyboard has its own nuances that need to be considered. For learning it there is even special keyboard simulators that help to learn to touch-type, which is very important when large texts and look at both sides, it's just a waste of time. The main thing is want it. Print fast - it's an art that requires study. For this there are special exercises that will make learning more efficient. In general, such simulators is very important today. That he can help make the job much easier and faster. [2]

My diploma is created to educate people typing on the keyboard. Using this application, the user can learn to type fast, using all ten fingers. You can learn from any of four languages: Kazakh, Russian, English and Turkish. For what would the user is not bothered by exercise, the program will be open every N minutes and closes after M minutes. Program duration and frequency user can change by himself, according to own convenience. After starting the program you will see seven tabs for switch between windows: Main page, Start, Test, Statistics, Settings, Instructions and Exit. If the user runs the program for the first time, he can pass the test to check its level, or simply click on the Start button and start from the first level. If the program does not start the first time, you can start with the START button and continue from the interrupted exercise. Clicking on the test, you will be taken to a new window where you can take the test to determine your level. After determining user’s level, he can go to the basic exercises, beginning at its level. On the top of the box is written number of level. Below is the text and textfield that you need to retype the text. The text is taken from the database according to the level. For creating database there is used class QSqlDatabase. The QSqlDatabase class represents a connection to a database. The QSqlDatabase class provides an interface for accessing a database through a connection. An instance of QSqlDatabase represents the connection. For checking correctness of pressed key there is used QKeyEvent class. The QKeyEvent class describes a key event. Key events are sent to the widget with keyboard input focus when keys are pressed or released. A key event contains a special accept flag that indicates whether the receiver will handle the key event. You should call ignore() if the key press or release event is not handled by your widget. A key event is propagated up the parent widget chain until a widget accepts it with accept() or an event filter consumes it.

At the bottom of the window is a virtual keyboard that shows which key should be pressed and by which finger. You can also re-start exercise by pressing the RESET button. At the bottom of the left side has tips if you make a mistake and do not know where. To see the progress, you should click on the Statistics tab. There are shown two graphs of black and red. The red color describes the error, black - speed. The X axis shows the number of days passed by the user, and the Y axis shows the number of errors for each day and an average speed of printing, that is how many letters have been printed in a minute. By direction of the graph up or down, you can determine your success. Ideally black graph should to strive upward indicating increased speed and red line should go down pointing to reduce errors. Graph is updated after each exercise is finished. Graph was be drawn used class QPainter. In fifth

window user can set or change settings. Chose preferred language among four languages: Kazakh, English, Russian or Turkish by mark only one of four QRadioButtons. After saving the changes program at all translates into chosen languages. Also in this window user can changes level from first till own level and upload other files to learn to type according to these files. There are settings of interval between closing and appearance of program, time duration of learning. After changing the settings user must press "SAVE" button. Then program install changes and transfer to start window.

After closing program, it does not stop working, it only turn to tray system on the taskbar on bottom side of window. In tray system user can change settings of interval and duration, open program or quit. In the last window of program user can read instruction of program, how to use it, properties of buttons and how does program work at all. Also in this window written how to sit while typing, which distance must be between user and monitor, what does virtual keyboard show and so on.

The project is not finished yet, but it has already implemented two of the four languages, virtual keyboard, automatic appearance, sound, timer, error checking, help on an error and the database.

Reference

1. http://it-event.ru/news/2060/
2. http://www.ixbt.com/soft/typing.shtml

УДК: 004.430

NoSQL: назад в будущее

Shaimardanova M, Bimagambetova R

Сегодня актуальны эффективные технологии создания специализированных решений с гарантированным временем реакции при обработке больших массивов данных, однако потенциал таких давно существующих подходов, как Cache, реализован еще не полностью.

Проекты класса XTP (extreme transaction processing, высоконагруженные приложения) предъявляют новые требования к СУБД — сегодня необходимы технологии, обеспечивающие низкую стоимость масштабирования и обработки больших объемов данных. К таким технологиям относят СУБД класса NoSQL, обещающие разработчикам высокую скорость внесения изменений в приложения, низкие затраты на масштабирование, инструменты обработки и хранения больших объемов данных, а также высокую скорость выполнения на относительно недорогом оборудовании. В базах NoSQL реализуется отличная от традиционной (одинсервер баз данных для нескольких приложений) парадигма, при которой одному приложению или модулю приложения предлагается отдельное решение по работе с данными. По своей сути архитектура решений NoSQL ориентирована на борьбу либо с большим объемом данных, либо с их повышенной сложностью.

Отличительные черты NoSQL — нереляционные модели данных, простые API или протоколы доступа, способность к горизонтальному масштабированию по требованию для некоторого набора операций на многих серверах, распределенное хранение данных, эффективное использование распределенных индексов и памяти для запросов, достаточно свободное обращение с такими незыблемыми для традиционных СУБД вещами, как транзакционная целостность. Общим для NoSQL-проектов являются компромиссы по отношению к взаимно противоречивым требованиям — например, уход от поддержки стандартных правил для обеспечения транзакционной целостности *ACID* (atomicity, consistency, isolation, durability, атомарность, согласованность, изолированность, долговечность) в пользу горизонтальной масштабируемости. Отказ от подобных требований был невозможен и является догмой для традиционных СУБД. Действительно, нельзя быть одновременно надежным, быстрым, распределенным и целостным, однако в ряде конкретных случаев возможны варианты.

Другой источник компромиссов — характер данных решаемой задачи. Именно тут важно требование к технологии, которая должна максимально использовать особенности предметной области. Например, если можно распараллелить обработку данных и использовать принцип *shared nothing* («без разделения ресурсов»), то нужно это эффективно применять как для хранения, так и для исполнения запросов. В этом случае надо строить модель хранения и распределения данных, которая опирается на эту возможность, однако в реляционных базах свободы выбора почти нет и