Issues in Information Systems
Volume 18, Issue 3, pp. 9-18, 2017

LOOKS DO MATTER: DEVELOPING AN ACCOUNTING SOFTWARE PROGRAM IN JAVA AND NETBEANS

Samuel Heredos, Walsh University, sheredos1@walsh.edu
Philip Kim, Walsh University, pkim@walsh.edu
Beth Vazzano, Walsh University, bvazzano@walsh.edu

ABSTRACT

This paper outlines the development process of an accountancy-budget based software application developed utilizing Java and the Integrated Development Environment, Netbeans. The report includes the methodology, results, and challenges experienced while attempting to develop a useful and aesthetically appealing software application for the accounting profession. The research suggests that form and aesthetics have a significant impact on the function and usability of a software application.

Keywords: Java, Netbeans, Usability, Design, Accounting Software

INTRODUCTION

This paper is the result of a year-long undergraduate Honors project to create a viable software program that could be used by accounting students and accounting professionals. The paper provides an overview of the development and decision-making process of creating an accounting software program written in Java. One of the objectives of this research was determine if form and style have an impact on the usability and design of a software application. The name of the software application is AccountBoost. AccountBoost provides numerous accounting and financial tools, but one of the more useful functions is it provides an ongoing live financial data. In addition to tracking accounts within the software, the program also provides tools such as investment feeds and currency converters. This paper provides context of this senior thesis project as well as the challenges and setbacks utilizing Java within the Integrated Development Environment (IDE), Netbeans.

The difficult part of developing an application is planning a product that can compete with what is already available to users. That is, how can a student create a product that increases efficiency for customers and is aesthetically pleasing for daily use? In 1992, QuickBooks hit the market and quickly became one of the market leaders in accounting software services. Soon after, Intuit, the developer of QuickBooks, released its Windows-compatible product and gained its market recognition between 1993 and 1995 (Iconis, 2014). Twenty years later, Intuit has now branched off and developed other financial service apps such as TurboTax and Mint. TurboTax is the tax preparatory aid and Mint is the mobile app for bill paying and budgeting money. Together these three products account for a majority of Intuit Inc.'s $4.2 billion in yearly revenue and its increasing stock price since the company went public in 1993 (Intuit, 2016). Three of the most integral elements are time-saving features, customer service, and ease of use (Angeles, 2016).

Some of the simplest applications have profited the most in download popularity because of a broadly accessible design. Rather than focusing solely on the functionality of the software program, it is equally as important the application is visually appealing and is intuitively designed (Young, 2015). Subsequently, the focus of coding this application was geared towards usefulness, aesthetics, and ease of use.

LITERATURE REVIEW

This review is focused on the Java programming language and how it can be applied to develop accounting/investment software, similar to the market leaders, in order to code a successful product. This study revolves around what other programmers are doing now to capture the market, what the market is moving towards over the next few years, and

what content needs to be included with accounting software to accommodate variance in demand. For example, there is a noticeable variation between the needs of a small business owner with employees and an individual entrepreneur tracking financial records (net balances between bank and credit card accounts, gain or loss on investment, tax returns, etc.). Through this analysis of accounting and computer science principles a fundamental list of accounting software requirements will be derived from various sources, with a focus on business owners, and average at home users.

Accounting Software

The first step in developing successful software is predicting a way to access a future user's needs. If you cannot provide desired functionality in the market, the product will not be a success. Many applications are hit or miss in this respect because a majority of their success rests on the marketing and aesthetics of the application: logos, layout, and color schemes (Xu, Peak, & Prybutok, 2015). To remain competitive, needs analyses and aesthetics preferences are essential.

Software-as-a-Service (SaaS)

Another consideration is where the software will be hosted. It is inevitable that the entire software market is moving towards the cloud so accounting will make the transition alongside its counterparts in the next decade (Gubbi, Buyya, Marusic, & Palaniswami, 2013). Firms already are predominantly in the cloud for research software, both for tax and for auditing and accounting (Dowling, 2014). This is a key part in deciding which market a developer will attempt to access. Increasingly, software vendors are offering free versions of their products. This approach gives the users a free trial of the program, which is often limited to reduced access to premium content or tools (Girsch-Bock, 2015).

Another similar comparison would be to the recent emergence of the Office 365 online accessibility. Microsoft now pairs a storage service with its Office products and email accounts. Vendors are also releasing software-as-a-service (SaaS) versions of their flagship products to attract cloud-only users (Drew, 2015). SaaS expands the horizon for IT Support as well, because it allows the support team to access an account over the web. Remotely taking control of a private device is similar to remote desktop host software such as Chrome Remote Desktop (Chrome, 2017).

The Utilization of Java Coding Language

The Java coding language is the designated language for the development of this research project. This language was chosen because of the adaptability that it provides (Gong, 1997). It also has an extensive background and numerous updates, which have extended its abilities far past those of many basic programming languages. Java has been used to write many desktop applications and games such as Minecraft and RuneScape, which have captured a large part of the Windows and iOS gaming markets. Java first hit the market in 1995 after it was initially released by Sun Microsystems (Java FAQ, 2015). As a result of the high concentration of gaming and database implementation Java has released updates in each of the last seven versions which provides new options for aspiring programmers.

Java Syntax

In an introduction to programming course, a computer application or "program" is a very broad term and is loosely defined because of the seemingly endless combinations of code that can be called a program. The "main()" method of a Java file is the outline of the entire program which executes functionality from one of multiple files. An operating system relies on multiple programs at different levels in the central processing unit (CPU) in order to operate correctly. In a Java-based program, one or multiple files are initiated in the calling routine of the IDE or the Command Prompt. The IDE and the Command Prompt both run the code sequentially to perform a task. The Command Prompt is pre-installed in the system on a Windows device (referred to as the Terminal on OSx devices). On either type of device, it is the direct link to communicate to the system what the administrator would like to do. Technically any user can use this content, but it is usually locked for administrator use only. Alternatively, the IDE is used to develop the code, save the file, and hold instructions for repeat commands.

In Java, there are methods, classes, and files all of which combine to make up the shell of the program. Additional code is developed by the programmer to fill the shell with different tools in the designated syntax. A frequently asked question in the computer science curriculum is, "what exactly is a program?" The program or the written language of

the program is a syntactically designed guide for the system, which is compiled and run with the purpose of executing the exact order of operations found within a file. Furthermore, the key to defining a program is its purpose. A computer program tells a computer what to do. Specifically, it provides a sequence of steps and instructions to complete a task (Horstmann, 2013). The development process in any coding language has a certain syntax, format, and queue, which executes each new section of the program. In Java, an object-oriented programming language, each file or "object" is named based on the class that it contains. Classes are used to separate tasks and to store various data, which all are sub-tasks of the task that will be completed in the "main class." Beginners find it most helpful to recognize that computer programming is separated into numerous sub-tasks, which are then connected by an interface to ensure compatibility. Additionally, the source code of many programs is often sealed after the finalization of development so no one can make an exact copy or tweak changes without consent of the developer. Privately withholding the source code ensures the marketability of the product which subsequently generates high revenue for the firm. This also limits copyright problems.

Major Accounting Statements

One basic accounting statement that is essential to any accounting software's repertoire is the income statement. The income statement is a list of earnings used to summarize the profit-generating and loss activities that occurred during a specific reporting period (Spiceland & Sepe, 2011). The basic format for an income statement is revenue less cost of goods sold equals gross profit. Then gross profit less operating expenses equals operating income. From there other items are added or subtracted such as gains or losses from components such as discontinued operations and extraordinary items. After applying the tax rate to the resulting income before taxes, the net income is determined and is shown at the end of every income statement. In addition to the other required financial statements, the process of categorizing different cash inflows and outflows is found within the cash flow statement. The four categories in the statement are cash flows from operating activities, cash flows from investing activities, cash flows from financing activities, and cash flows from noncash activities. The difficulty in preparing a statement of cash flows is in identifying in which category each cash inflow or outflow belongs.

Looks Do Matter

Design principles are often utilized by graphic designers, but are too often ignored by programmers. Visual "harmony" is when the arrangement or parts, such as images, colors, and buttons are in balance with one another (Lauer & Pentak, 2002). When a website is more pleasing to the viewer, it is said to be in harmony or balance. And when a website is out of balance, the viewer observes the site or application to be boring or disorganized. Even though the functionality of the website or application may be identical, if the look and "feel" the site is off, users will be discouraged from using the application (Moshagen & Thielsch, 2010).

With website design, users respond to specific design elements. Therefore, the user may feel a sense of satisfaction when website colors are appealing, or when a graphical design elicits enjoyment or excitement. In addition, it is important that website design meets the needs and sensibilities of the user (Cyr, 2009, par. 6).

As competition for mobile applications increase, application programmers must not only design software based on user requirements, but also on how it looks. Thus software design must go beyond how well it completes a function, it must also consider its perceived usability. Not surprisingly, researchers have found a correlation between perceived usability and how aesthetically pleasing a program is to the user (Lauer & Pentak, 2002; Tractinsky, 1997). Lindgaard, (1999) found that color is the best predictor of whether users found the program useful or not.

To design an application and to appeal to a broad range of users, it is important to review the software programs already in the market. We not only analyzed the function of the program, but also the form: the physical and psychological attributes of the programs. The "psychological advantage" or the elements of a program that each individual developer pinpoints for themselves provide benefit because of the spontaneity and creative nature from which the application was derived (San Martín & Herrero, 2012). As the modern application market proves, new developers frequently release updates or new products that sell better than the previous versions because they provide a new feature or better design. There are multiple psychological connections associated with color schemes. This applies to colorful items that are frequently included in applications. This could include software logos, user-interfaces, menus, and home screens. Colors that are pertinent to this research include:

White is associated with purity, cleanliness, virtue, happiness, sincerity, and safety. Orange is an energetic and vibrant color often associated with fun, happiness, energy, warmth, ambition, excitement, and enthusiasm. It can also be used to communicate caution. Green has a harmonizing, balancing effect. It is associated with growth, health, nature, wealth, money, calmness, masculinity, generosity, fertility, envy, good luck, peace, harmony, support, and energy (Bowman, 2014, p.2).

As an example, some Big-Four accounting firms implement a soft green default setting on their employees' laptops. The soft green color screen enhancement is auto-adjusted by the Windows operating system. The "green scheme" is an innovative way to save the user's eyes. For an employee who looks at a screen most of the day, soft green provides less strain than the default screen settings. Some components of software coding can become a win-win situation if the developer puts in the time to include items such as the soft green hue to the screen. Not only does this new possibility on the computer-science and eye-health spectrum allow for improved physical health, it can also become an unintended base for color branding of the product. In the financial industry it just so happens that green is often associated with wealth. Since wealth management and boosting financial status are two of the main goals of this project, this will be one of the key color schemes within the application. Formatting and layout of these tools, which includes font, color scheme, placement in the application, and depth within the menus, all correlate with some type of psychological stimulation (Brady & Phillips, 2003).

METHODOLOGY

The framework for the development of AccountBoost is focused around the utilization of Java coding skills and the Integrated Development Environments or IDE: NetBeans. These programming tools were used to develop a fully functioning accounting software program for the everyday user. That is, an individual who is not a financial services expert. As such, implementing tools like an easily accessible financial calculator and an up-to-date dynamic stock market tracker are functions critical to ease of use.

Java was the programming language of choice because of its adaptability, currently concentrated usage in the market for business applications, and its background of knowledge from the student and advisor perspectives. The development process began with the naming of files, initiation of variables, and the outline of comments within the IDE. The framework created by the outline was then populated with the source code. Sections of this code required techniques that were unknown to the developer and had to be further explored to determine how to run the software.

Additionally, different usage of methods, classes, and file structure were evaluated to create a more efficient use of the loops and instruction set. It is unknown how much memory the users will have available on their systems, so developing a program with maximum efficiency was explored to increase the overall user experience. As with many disciplines, there are multiple techniques to achieve a similar outcome, so it is important to consider other variations of code syntax to perform the same functions. The three most important elements of this study were to design the program with the appropriate functionality, optimize code efficiency, and design the software to be easy to use. The scope was limited to budgeting and accounting-related functions. This did not include security valuations or advice on investment decisions.

Generic Programming to Increase Compatibility

Within the outline for development, three goals will broaden the compatibility of the product. The first goal is to write the code as broadly as possible with the integration of Java and NetBeans inspired interfaces. Generic programming allows the developer to re-use the same code with different inputs (Horstmann, 2013). When code is written generically it can lead to expanded accessibility from other classes. When planning for a future product and future updates these are tasks that can be applied at the beginning for increased lifetime efficiency. The secondary goal is to have a server login on all levels. The application will not be open for use by anyone who can access a system which has the software installed. This access will be restricted by a username and passkey which will be designated by the user when the software is first installed. The third goal is to make two different versions of the software: one full and one limited. As in many modern day applications the users who have installed the limited version will be able to view the full version capabilities, but not use them. This will still allow the user to get the most out of the free version, but

provides an opportunity for the user to upgrade to the premium version of the software through software upgrades. These software upgrades will generate revenue from the product's use.

Integrated Development Environments (IDEs)

The development of accounting software with a focus in Java can technically be written in any text file and can be run by a Java compiler, but IDEs allow for an aesthetically pleasing experience with many tools to offer programming support to the developer. Two useful IDEs to the programming world are JGrasp, which was developed by Auburn University, and NetBeans, which was developed by Sun Microsystems (Levitin & Mukherjee, 2003). JGrasp provides a compiler, debugger, Run I/O, Warnings, and Interactions. It also has regular expression testing as well as garbage collection. NetBeans has all of the abilities of JGrasp, but differs in that it has src generation. That is, it generates source code based on the placement of items on the screen, similar to laying out PowerPoint slides. The IDE then generates the code to map the technique that the developer used and adds the code to a Java file. From there modifications can be made to update the file's functionality. IDE's are one of the most highly used programming tools because of the adaptability that is available to the programmer. Visualizing code can help the beginner programmer to write a far better source code product and the IDE's provide the platform (Kordaki, 2010).

Accounting Applications

One of the keys to a successful accounting software application is the ease of the user-interface that it provides to a client. The basic and complex accounting equations that are required under the Generally Accepted Accounting Principles (GAAP) are what will actually be found within the code. Ratios for corporations can be calculated to find key figures about the current financial position of the company. For example, to find the current ratio of a corporation, which measures a corporation's liquidity (the ability to pay short term debt); this value can be found by dividing the company's current assets by its current liabilities (Spiceland & Sepe, 2011). In programming terms this can be done by creating a method with a name and parameters of: calcCurrentRatio(double cA, double cL). The method would return the floating-point value of the calculated answer. In the NetBeans IDE, there would be a tab to calculate ratios. Within that tab would be found a list of ratios and the user could select the current ratio option. From there the user would be navigated to a window to type in the values of the current assets and current liabilities. The values typed in the text boxes are passed through the calling routine to a method with the syntax of calcCurrentRatio(currentAssets,currentLiabilities). From there, the liquidity of the company would be calculated by the system and would be returned in a pop-up window where the user could choose to save the returned value.

Another form of program that could be written to satisfy the market's needs is one that could automate journal entries for the system by prompting the user for only the most minimal input. For example, in construction, accounting journal entries are extremely important to record and balance the cost of construction versus gross profit, cash collection, etc. The reason this is so important is because when recording revenue by either the percent completed method or by completed contract method, the company has to conform to American GAAP regulations in order to legally record a sale. If this is done incorrectly, either by recording the wrong numbers or at the wrong point in the sales process, the company could be vulnerable for misrepresentation findings in future audits. As a result, a journal entry program would become an essential part of many different organizations' financial structure. An example of this type of program would be a Java file that contains a method called printConstructionEntries(int constructionInProgress, int accountsReceivable, int cash, int revenue). Four integers are passed to the method as explicit parameters but in construction accounting nine different integers are required for one journal entry. Each of the remaining five integers can be derived from one of the explicit parameters, therefore only the ones that are needed are input by the user in order to simplify the journal entry process. Simplicity is key in this context. Automated journal entry methods introduce a new level of simplicity.

RESULTS

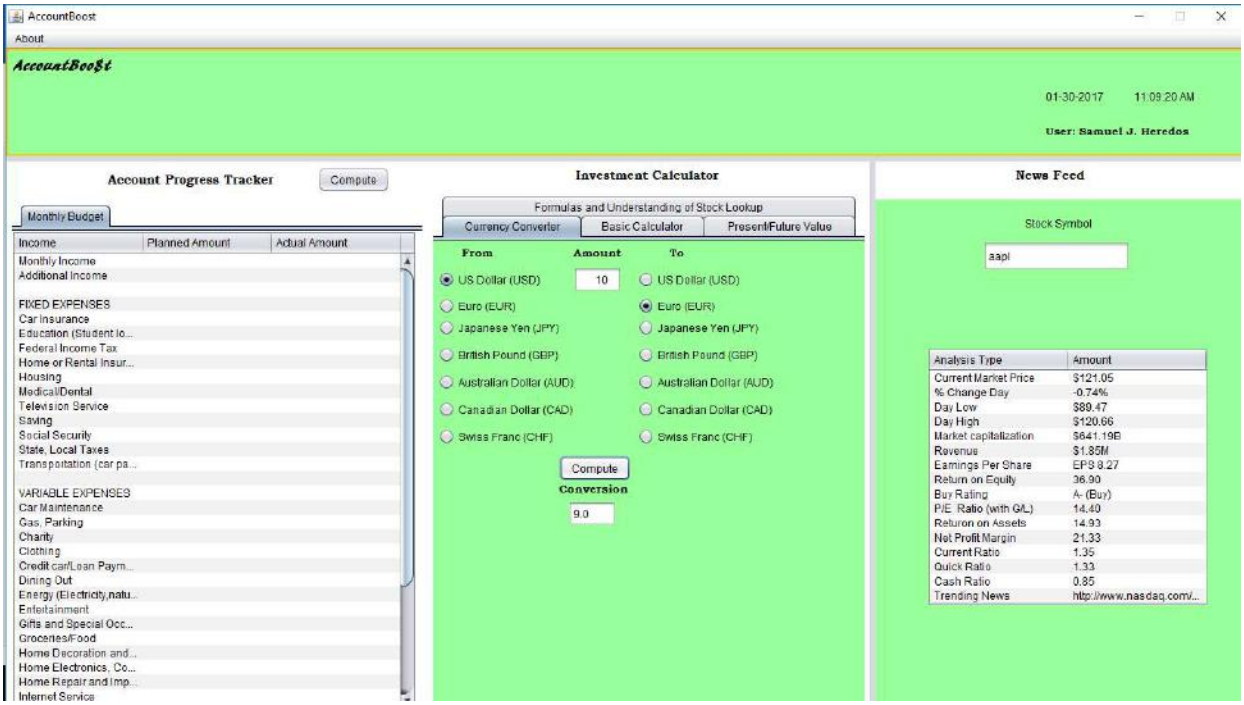

Figure 1. AccountBoost User Dashboard

The finished product is named "AccountBoost" (figure 1). Its purpose is to help users track their account balances, maintain a budget, and calculate difficult and challenging values using the built-in tools. As a result of some early feedback from beta users, several additions were made to the software. The most prominent feature is the "news feed" to aid the user in tracking press releases about important parts of the world economy. This idea then evolved to a stock symbol lookup function. This aspect of the dashboard runs loops that pull data from all over the web and store the data within arrays generated by the software. Because they are in real-time, the updates provide the most accurate information every time the user searches for his or her investment.

Monthly Budget

The monthly budget section of the thesis allows the user to plug in his or her budgeted (and later, actual) monthly income and any additional income that was earned from investments. The user can then input his or her budgeted (and later, actual) expenditures by choosing them from the list of typical expenses. It gives users an accurate representation of their budgeted and also their actual net cash balances. This application will be useful to individuals as it is common to overlook the reality of a budget and then spend more than you have (Schicks, 2010).

News Feed and Lookup

To accomplish the investment newsfeed section of the software, extensive troubleshooting and trial and error was required to achieve this difficult task. In the planning stage, the following methodology was used: allow the user to input stock symbols into the table, for example, AAPL (Apple), LUV (Southwest Airlines). Take the stock symbol and cross-reference it with an online database such as Yahoo Finance, Google results, or an actual investment tool website such as MarketWatch. The remainder of the functionality was supported by referring to the Nasdaq Dozen. Nasdaq is a reliable source for information regarding investments because it is "the second-largest exchange in the world by market capitalization, behind only the New York Stock Exchange" (How to Analyze a Stock, 2016). It also advises investors in regards to decision making and growth chart analyses. The goal of this part of the software is

accomplished in the code by a concept called web-scraping. Downloading data off the web, also known as web-scraping, allows the user to input data, run a search of the web which is usually limited to a certain area (so in this case to Yahoo Finance, Google, and MarketWatch), and then return the data back into another source, AccountBoost. The return value is a string or a sequence of characters, which contains the HTML.

Figure 2. AccountBoost Stock Symbol News Feed

Web Scraping (JSoup)

The process of searching for elements on the web as tags and classes was a new concept to the developer so the software only began to take shape after the preliminary research and instructional videos. The developer used the JSoup data library which imports the ability to code this type of task in NetBeans. In the runnable version of the software, the user can simply log in and reference the table section in the right hand column. From there the user is able to type in the stock symbols and 16 other financially relevant data points provided by Nasdaq. This means the user can use the multiple tools in the user dashboard to conduct calculations while researching real-time financial data (figure 2). The middle pane in the software contains an investment calculator layout. The different investment ratio calculations, time value of money (present and future value), basic calculator (if needed) and the currency converter (figure 3) are very simple to use but perform the complex calculations that many people have trouble comprehending. Additionally, the software provides explanations through text boxes with labels on each tab, which guide the users as they are navigating the software.

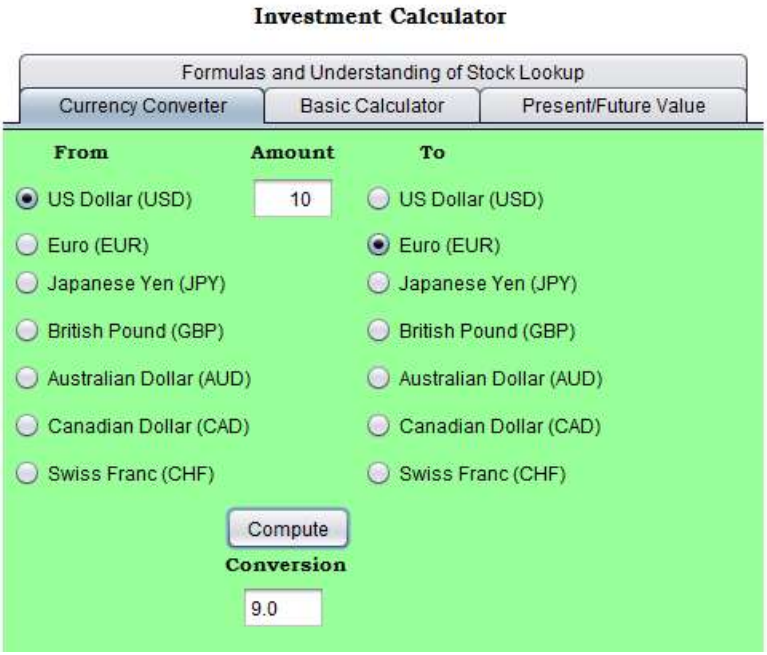

Figure 3. AccountBoost Currency Calculator

CONCLUSION

While AccountBoost is fully functional, it is not commercially available yet for download. Although the research objectives were met, there were some limitations to this study. First, due to the time and resource limitations, this research project was developed during the course of an academic year and utilizing free online resources to code and compile the java application. It should also be noted that all coding was completed by the primary investigator. This may have affected the end result as these development activities were conducted in conjunction with a full course schedule. For future similar research projects, it may be helpful to incorporate a team-based approach to mobile application design.

The next step for this research project would be to conduct a beta usability test for accounting and finance-related students. After collecting data and feedback from the beta users, subsequent steps would include releasing the program in the open market to collect further user reactions and evaluate its economic viability. In order to take this software to market, the developer would need to buy cloud storage space for a database so that more than a few user profiles can be stored. Making the software commercially available went beyond the scope of this thesis project; however, it provided some significant learning opportunities for this developer and could be used as a case study for future similar projects that use Java and Netbeans.

REFERENCES

Angeles, S. (2016, January 4). *QuickBooks Online review: Best small business accounting software.* Retrieved March 10, 2016, from http://www.businessnewsdaily.com/7544-best-accounting-software-small-business.html.

Bowman, D. (2014). *The Psychology of Color in Web Design: Part 1.* Retrieved from http://www.jimdo.com/2014/07/25/the-psychology-of-color-in-web-design-part-1/.

Brady, L., & Phillips, C. (2003). Aesthetics and usability: A look at color and balance. *Usability News, 5*(1), 2-5.

Chrome. (2017). *Chrome Remote Desktop* retrieved from https://chrome.google.com.

Cyr, D. (2009, May 1). *Emotion and website design:* The Encyclopedia of Human-Computer Interaction, 2nd Ed. Interaction Design Foundation. Retrieved from https://www.interaction-design.org/literature/book/the-encyclopedia-of-human-computer-interaction-2nd-ed/emotion-and-website-design.

Dowling, C., & Leech, S. (2014). A Big 4 Firm's use of information technology to control the audit process: How an audit support system is changing auditor behavior. *Contemporary Accounting Research, 31*(1), 230-252.

Girsch-Bock, M. (2015). Nonprofits need strong accounting software for effective management. *CPA Practice Advisor, 25*(4), 8-12.

Gong, L. (1997). *Java security: Present and near future*. IEEE Micro, 17(3), 14-19.

Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M. (2013). Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems, 29*(7), 1645-1660.

Horstmann, C. S. (2013). *Big Java: Late Objects.* John Wiley & Sons, 2013.

How to Analyze a Stock: NASDAQ Dozen Stock Analysis -. (2016). Retrieved from http://www.nasdaq.com/investing/dozen/.

Iconis, C. (2014). *IconisGroup - Certified QuickBooks Online ProAdvisor.* Retrieved March 11, 2016, from http://iconisgroup.com/officially-a-certified-quickbooks-online-proadvisor/.

Intuit. (2016). *Company fast facts.* Retrieved March 11, 2016, from http://www.intuit.com/company/fast-facts/.

JGRASP. (2015). *JGrasp history.* Retrieved November 3, 2015, from http://www.jgrasp.org/.

Kordaki, M. (2010). A drawing and multi-representational computer environment for beginners' learning of programming using C: Design and pilot formative evaluation. *Computers & Education, 54*(1), 69-87.

Lauer, D. A., & Pentak, S. (2002). *Chapter 5: Balance. Design Basics.* (pp. 75-98). Australia: Wadsworth.

Levitin, A., & Mukherjee, S. (2003). *Introduction to the design & analysis of algorithms* (p. 576). Reading: Addison-Wesley.

Lindgaard, G. (1999). *Does emotional appeal determine perceived usability of web sites?* Hawthorne: University of Technology, School of Information Technology.

Moshagen, M., & Thielsch, M. T. (2010). Facets of visual aesthetics. *International Journal of Human-Computer Studies, 68*(10), 689-709.

San Martín, H., & Herrero, Á. (2012). Influence of the user's psychological factors on the online purchase intention in rural tourism: Integrating innovativeness to the UTAUT framework. *Tourism Management, 33*(2), 341-350.

Schicks, J. (2010). Microfinance Over-Indebtedness: Understanding its drivers and challenging the common myths. Bruxelles: Centre Emilee Bergheim, Solvay School of Business, CEB Working Paper, 10, 048.

Spiceland, J., & Sepe, J. (2011). *Intermediate accounting* (6th ed., combined ed.). New York: McGraw-Hill Irwin.

Tractinsky, N. (1997). *Aesthetics and apparent usability: empirically assessing cultural and methodological issues.* Paper presented at the 1997 Chi conference. Retrieved from: http://www.acm.org/sigchi/chi97/proceedings/paper/nt.htm.

Young, S. (2015). *15 tips for succeeding as an independent app developer.* Retrieved March 11, 2016, from https://www.entrepreneur.com/article/247622.

Xu, C., Peak, D., & Prybutok, V. (2015). A customer value, satisfaction, and loyalty perspective of mobile application recommendations. *Decision Support Systems, 79,* 171-183.