

The FunWritr Language & Literacy Playground: mashing up the web for English acquisition, language growth, and Exploration

Justin Olmanson* Jaejin Lee Cesar Navarrete Angela Pan Wong Alon Farchy Patrick Day

Language Learning & Technology Research and Design Group

Instructional Technology Program

Department of Curriculum and Instruction

University of Texas at Austin

*olmansju@gmail.com

Abstract: As more online technology applications make their content available to developers, the opportunities to create both novel and supportive tools for creation, elaboration, and display of written text by English language learners in the middle elementary school grades and beyond expands greatly. This paper describes the features and process of development of the FunWritr application. FunWritr is an educational mashup that supports student language exploration and open-ended writing. By using existing natural language processing tools such as grammar parsers (NLTK), semantic ontology databases (WordNet), and image collections (Flickr) FunWritr affords users the opportunity to view visual representations of the words and phrases they write and the word families to which they belong.

Sections: Rationale & Background, Design Description, Development Description Server & Flex Side, Timeline?, Collaboration Description, Research Group Description, References

Development Rationale and Background

As more online technology applications make their content available to developers, the opportunities to create both novel and supportive tools for the composition, elaboration, and display of written text by English language learners in the middle elementary school grades to adult expands greatly. Previous attempts at supporting open-ended student writing via stand-alone software or the web have tended to suffer from a need to anticipate all possible communicative directions users might take and provide scaffolds for these. Such an endeavor often becomes an expensive exercise in broad-spectrum or anticipatory content creation or allocation. Consequently, designs for writing software often end up relying on a set number of prompts or themes to constrain content for the

sake of decreasing the range and amount of supports (Boris, n.d.; *Storybook Weaver Deluxe*, n.d.; *Ultimate Writing & Creativity*, n.d.). Conversely some writing applications seek to maintain open-endedness sacrificing the opportunity to provide rich multimedia supports in favor of modest, text-based scaffolds (Ashley, n.d.; *Microsoft Word*, n.d.; Tiffany, n.d.; *WhiteSmoke*, n.d.) which may fall short of the needs of many beginning and intermediate English language learners, especially those in the middle elementary school grades. Finally, the handful of mashups available in the domain of writing are either targeted for very young users still learning the English alphabet, such as AlphaLearnr (harikrish, n.d.), or programmed in to add aesthetic value instead of pedagogical support as seen in digiPoem (Elliott, n.d.).

A mashup is a web-based or networked application which leverages existing content in a value-added way. Mashup creators use special information requests via APIs to reconfigure content. For example data from a website of apartment listings might be juxtaposed with neighborhood crime rates on a Google map –thus giving out-of-town apartment seekers a visual way of identifying available apartments in safe areas.

Up until the emergence of unbundled Web 2.0 APIs, such mashed-up (Liu, Horton, Olmanson, & Wang, 2008) applications were merely the stuff of conjecture. The emergence of a range of content repositories make a constellation of writing supports in multiple modal channels and at varying levels of English language familiarity possible –allowing users to negotiate meaning, notice salient syntactic forms and semantic structures (Chapelle, 2003) in less constricted ways. These affordances make it possible for applications to allow user to explore the dynamic construct of English down myriad avenues while still benefiting from the types of linguistic scaffolds previously only enjoyed by users working their way through scripted content.

We, a group of students in the Instructional Technology Department at the University of Texas at Austin, wanted to explore the potential of mashups in education and so endeavored to create one which might inspire others, help students, and expand the process of writing beyond the confines of memorized vocabulary (Cook, 2001; Wood, 2001), spelling conventions (Seymour, Aro, & Erskine, 2003), and grammar rules and into a realm of expansive picto-ontological worlds where elaboration, linguistic precision, collocation, and play are just a click or mouse-hover away. A survey of existing writing software led us to conclude that they either supported writing by either a) recreating the blank page with supports geared for native speakers, or b) funneled the writer to specific exercises designed to improve discreet writing skills. We wanted to create a sort of “magic window” that takes the user into the shifting, variegated meaning of the words they type and the stories they create regardless of the thematic direction their language exploration takes. We wanted the language learners themselves to plot their course without impinging upon the level of proffered support –in hopes of increasing the generativity of the interaction within a seriously playful (Rieber, 1996) environment. We named our experimental space for language exploration and personal expression FunWritr –adopting the playground metaphor as we felt an open-ended, support-rich, online writing application would have much in common with a playground -making language exploration for learners at many different levels of cognitive development and English language ability possible.

Just as playgrounds have equipment for organized, rule-based activities such as basketball, soccer, and four-square; the application can be incorporated into instructional activities that focus on spelling conventions, grammar acquisition, vocabulary building, and prompted narrative and expository writing –thereby meeting the school and classroom teacher at their level of need and interest (Figure.1).

Design Description:

The interface design of Funwritr was influenced by the fact that our initial users will be elementary-age children. We worked with a teacher at a local school, observing her students using other writing software and doing a range of literacy tasks. The iterative design is web-based, and uses the relationship between words and text in a freeform "playground" structure. Depth and light were used to establish a sense of open-ended space within the established boundaries of the window. Variations on muted primary colors connect various functions visually; blue is used mainly to connect text and image results, yellow for word explorations.

The design process took several iterations. As our understanding of the usability and accessibility parameters developed, the design team pruned and honed specific functionalities. The goal was to balance the addition of features that are appealing and educationally valuable with the design principles that make the

application simple and intuitive for a second grader to use. For example, earlier design versions placed a larger emphasis on what could be done with the images that were brought back; the user could flip through a carousel of image search results, select specific images to go with words, or put the images in a lightbox. Though interesting and attractive, the team decided that these features would be distracting to the fundamental purpose of engaging the user to write and explore language.

As the focus shifted more towards the interactivity of writing, the interface evolved in that direction as well. The input box went from a singular line to a multiline text field. The writing space and image area were flipped so the continual flow of written text could flow upwards. Attention to subtle contrasts in size, opacity, and depth allow for the alleviation of visual clutter and bring to the foreground information that could be helpful for the user's writing process.

Technology Description

By using existing natural language processing tools such as grammar parsers (NLTK), semantic ontology databases (WordNet), and image collections (Flickr) FunWritr affords users the opportunity to view visual representations of the words and phrases they write. The FunWritr user interface is Flex-based with a CherryPy backend and sqlalchemy database. As learners produce text it is parsed and semantically-based, visually-rich content is returned. FunWritr is image-centric, and ontologically organizes aspects of language.

The flex side of the FunWritr project required something that was stable, flexible, and performed quickly enough for our purposes. To achieve this, each component of the project was modularized and connected to a central control module that serves as the backbone and guide of the browser-side application. The primary components include the input box, image bubbles, and server communication. All three modules communicate exclusively with the control module, which allows for easier programming and debugging while providing a more deterministic, stable control flow.

The input box has several functionalities. Firstly and most importantly, it needed to alert the module of spacebar presses so that it could send the newly written text to the server-side portion of the application via PyAmf to be processed. Secondly, it needed to highlight the text currently being displayed by the display aspect of the program and manage the highlighted text even as it is being edited. Finally, the Input Box was meant to display which part of grammar each word is underneath the words themselves.

An image bubble contains a word or phrase and relevant images. The images are dynamically loaded from Flickr and displayed in a semi-randomized manner. A separate container class is in charge of these bubbles, positioning, adding, and removing them dynamically as necessary. Styles for all graphical components are handled in a separate module to reduce clutter.

The flex application is connected to a python backend that handles the parsing and analyzing of input text. After much debate, a protocol was established between the flex application and the python backend that would allow for a rich flow of information between the processes while minimizing performance costs. Some tasks, such as word disambiguation invokes a significant performance cost, and therefore is done as infrequently as possible. Word and phrase information is cached within the flex application and stored in the python database to improve response time. Finally, session and transaction information is logged to organize and coordinate messages between Flex, the python backend, and the database.

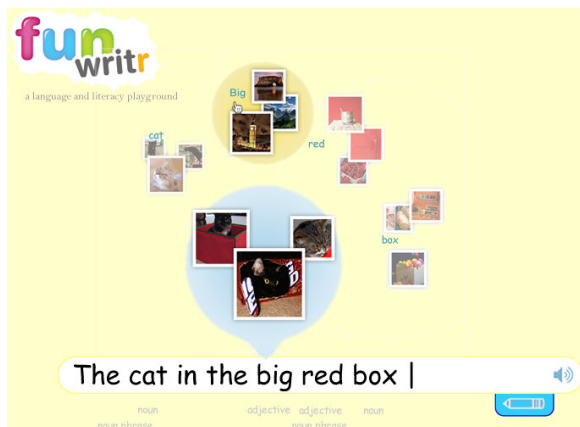


Figure 1 Screenshot of FunWritr in action with the user about to explore the word 'big'.



Figure 2 Screenshot of the language exploration view for the word 'big'.

Just as there are safety features and regulations on physical playgrounds, FunWritr was designed to employ several strategies to ensure the images returned are appropriate for the classroom environment. The use of dynamic, open-ended data sets such as the 230,000,000+ images in the Flickr database creates the possibility for potentially inappropriate content to be displayed within the context of the application's use. While some risk is inherent when working with dynamic third-party content we have taken several measures to mitigate these risks. FunWritr filters the 1200 words most likely to return such content –substituting equivalent inoffensive words in their place. Additionally our image sorting algorithm is set to relevance and not interestingness further reducing the potential for objectionable content and the request for images is set to 'safe content only'.

Issues and Project Trajectory

This project required a number of students working in parallel and in small increments –often in an ad hoc fashion between classes, projects assignments, work, and meetings. This piecemeal coding was made possible via weekly meetings and the use of versioning software called Subversion.

Collaborative programming is one of the major trends in global software development area (Lanubile, Ebert, Prikladnicki, & Vizzaino, 2010) . The development team drew upon collaborative programming to help manage the efforts of a team consisting of three computer programmers, a web designer, two instructional technologists and a project manager. For the effective communication and monitoring the developmental flow of Funwritr, Subversion (SVN; <http://www.subversion.tigris.org>) was used. Subversion is popular open source version control system and SVN has centralized architecture which allows individual user can check in and out with client application from a server.

This process, although seemingly simple, can become an issue on to itself. Much of the open source shareware is platform sensitive. Each of PC and Mac users in the team have used different Subversion systems to access the project and the developmental software such as Flex3 and Flash CS4 caused many troubles when one tried to open the files which was downloaded through SVN within the developmental software. This interface can add elements of frustration to the project as files can get dropped or access is impeded.

Research Group and Management Plan

The project work is the focus of the Language Learning & Technology Research and Design Group whose membership is made up primarily of individuals living in Austin TX, most of whom enjoy some affiliation with the University of Texas at Austin's Instructional Technology Department. More information about the FunWritr project is available at FunWritr.com

References

- Ashley, T. (n.d.). *Writer's blocks3*. Ashley Software. Retrieved May 1, 2010 from <http://www.writersblocks.com/>
- Boris, J. (n.d.). *Easy Writer*. Softwareforstudent.com. Retrieved May 1, 2010 from <http://www.softwareforstudents.com/SoftwareMainPage.htm>
- Chapelle, C. (2003). *English language learning and technology: Lectures on applied linguistics in the age of information and communication technology*. Language learning and language teaching. Amsterdam: John Benjamins Pub.
- Cook, V. J. (2001). *Second Language Learning and Language Teaching* (3rd ed.). London: Arnold.
- Elliott, J. (n.d.). *digiPoem*. Writr. Retrieved May 1, 2010 from <http://www.dudeyjon.com/digipoem/index.html>
- harikrish. (n.d.). *AlphaLearnr*. Retrieved May 1, 2010 from <http://www.rapidmonkey.com/alphalearnr/>
- Lanubile, F., Ebert, C., Prikladnicki, R., & Vizcaino, A. (2010). Collaboration tools for global software engineering. *Software, IEEE*, 27(2), 52-55.
- Liu, M., Horton, L., Olmanson, J., & Wang, P. Y. (2008). An Exploration of Mashups and Their Potential Educational Uses. *Computers in the Schools*, 25(3/4), 243-258.
- Microsoft Word*. (n.d.). Microsoft. Retrieved May 1, 2010 from <http://www.microsoft.com/office/>
- Rieber, L. (1996). Seriously considering play: Designing interactive learning environments based on the blending of microworlds, simulations, and games. *Educational Technology Research and Development*, 44(2), 43-58.
doi:10.1007/BF02300540
- Seymour, P. H. K., Aro, M., & Erskine, J. M. (2003). Foundation literacy acquisition in European orthographies. *British Journal of Psychology*, 94(2), 143. doi:Article
- Storybook Weaver Deluxe*. (n.d.). . The Learning Company. Retrieved May 1, 2010 from http://www.kidsclick.com/descrip/sbw_deluxe.htm
- Tiffany, S. (n.d.). *Icon Poet*. USA. Retrieved May 1, 2010 from <http://www.iconpoet.com/>
- Ultimate Writing & Creativity*. (n.d.). . The Learning Company. Retrieved May 1, 2010 from http://www.kidsclick.com/descrip/ultimate_writing.htm
- WhiteSmoke*. (n.d.). . Wilmington, DE: Whitesmoke Inc. Retrieved May 1, 2010 from <http://www.whitesmoke.com/>
- Wood, J. (2001). Can software support children's vocabulary development? *Language, Learning & Technology*, 5(1), 166-201.