Address API – User Testing

October 29, 2016

Melissa Fraley and Jenny Stanhope

Address API, is a data access site, which is accessible through an Application Program Interface (API). The anticipated users, for this database, are computer programmers who want to access a database for more information on a particular address. The database's function is to return attributes on variables such as addresses, neighborhoods, council districts, and zoning in reply to queries generated by computer programs. Currently, the site only contains attributes for addresses in Missouri, however, in the future it will include information for addresses in Kansas as well. Since open data sites vary, regarding the types of attributes they include in their databases, this site is attempting to give researchers and software developers a place in which they can find all the relevant attributes they need for their project. A primary goal or function of this application is to allow programs to utilize address information when coding the "back-end" of their projects. The computer languages used on this site includes PHP and Python, among others. User testing will be performed with computer programmers, to obtain feedback on the site in general and on how it might be improved.

Methodology

Two experienced computer programmers (10+ years of programming) were utilized as participants in this research. The participants were asked to go to address-api.codeforkc.org and provide feedback about the site in general. They were given two tasks to complete. In addition, an unstructured interview format was used while participants explored the site.

Participants were asked a series of open and closed questions throughout the session, prompting them to expand on their thoughts, provide feedback, and clarification. Participants were also prompted to provide feedback, about the potential uses and limitations of the site. They were asked about using the site in the "back-end" coding of their future projects. Audio and video recordings were made for all interviews. The recordings were transcribed, time-stamped, and coded for relevant information. The recordings were then analyzed to provide feedback on usability and recommendations for improvement of the site.

As the participants attempted to complete the tasks, they were prompted to speak their thoughts aloud and move the mouse around as they explored the site. Feedback about their interaction with the site was encouraged as they explored the site freely. In Task 1, the participants were asked to look up the following address: 7401 Main. In Task 2, the participants were asked to look up a second address: 210 W 19th Ter. The first question, posed to the participants, asked why they thought the developers of this site had difficulty getting numbers and directions (i.e., east, west) to come up in the address field. The second question asked participants was what they thought about the addresses only being available for Missouri and Kansas. Additional questions included: what other information they would like to have included in the site; if they have any suggestions in regards to how the site is organized; if they have any other comments or suggestions about the site; and if they would use the site on the backend of their projects.

Results

It was noted, by both participants on Task 1, there was no feedback provided after entering the address into the address field. This was problematic because when they clicked on the address that appeared below the address field there was no indication at that point they also needed to click the submit button. Participants thought the program was not working and stated to manipulate the address before the program had a chance to search for the information. Due to the empty space below the address search field there was no indication of what the page would look like once the appropriate address was submitted. If there were feedback to indicate the program was searching of if there were fields below you would expect to populate once the search was complete for a particular address, participants might not have assumed the address search field did not work properly. In addition, both participants expressed the need for more information to be provided on the home screen about how to enter the address exactly the way the program requires. For example they require information on how to enter street directions (i.e., W or W. or West). They need to be very precise when entering address information into their programs when they send a query to an API to check address for them, especially when they perform bulk queries. In addition, both participants thought it looked like the program finished populating the address search field for them. They both stated this would be very helpful to them if they were looking up an individual address. Upon looking for information on the site which would give them the precise requirements of the program for entering an address into the search address field, they both discovered that the API tab did not work. They found some helpful information in the DB Field Column under the Attributes tab. However, they stated this column would be more useful if it were placed on the home screen; either to the left or right of the columns already on that page. They also

thought the information provided in the DB Field column essentially, "states the obvious," in that it does not provide them with enough information. One participant stated, it looks like there was some geocoding but he needed to know what system was used for the geocoding. Information the programmers need to include in their coding, might be too lengthy to incorporate in your standard DB Field column, however, it is necessary information they need to have access to. The programmers felt this could be addressed by providing a link to the information (i.e., a link to the Geocoding standards). A participant also questioned what data type would be returned (i.e., a *floating point*). One participant, commented on the API returning data in *spreadsheet*. He stated most current programs use JSON or XML. It appears developers are using a *restful interface* but the programmers felt it should be explicitly stated on the site what type of interface they are using. When attempting Task 2, an *error message* came up for both participants. For participant two, the *error message stayed there even after he entered the* precise address into the field. It was apparent to the participant he had the precise address because the information on that address appeared below the address search field, however, the error message did not go away. Also on Task 2, two results came up for that address which included floor one and floor two. Having more than one result to choose from for the same address could potentially be problematic if their program is not able to choose which one it needs. The participant wants to know what he would get back in that case (i.e. will it send back an array of JCONs). The participants see some value in having an interface like this to look up an address if they have a "bug" in their coding. They would be able to go here and look up the exact way to put the address in. For programmers, every character has to be perfect in order for their program to find what it is looking for. The more work the

developer, of the API does doing artificial intelligence work on their end, the more useful the program would be for the programmer. Participant two stated that having all of that kind of work done by the developer of the API would be a major selling feature. The programmers guessed the reason the developers are having difficulty finding numbers and directions, "is that there is an official address that's in there that may not be in sync with what people are actually using in real life." When the programmers assemble a long query in JAVA it is possible they would be assembling this (address) in the code. He stated, it would be nice if they were able to copy the query their code generated and paste it in the API to see what it comes up with. They would bring it up in a utility that just does the query for them so they can see if it is working or if something is wrong with their code. One of the participants stated, "this is terribly not important to me as a programmer...that interface...what we're going to be writing programs that will be generating that URL." Final suggestions the participants would give to the developers included giving them more information in the form of links. The information they have access to in *GitHub* for this program was not organized in a way that would be useful to them and they felt that it was essentially what was supplied on the About page. They suggested again that the developers consider using JSON or XML, instead of spreadsheet because they are more standard practices now. JSON or XML are nice to have if they were able do bulk downloads of addresses. They both thought it is a great tool in terms of being an open source for certain purposes like for use by the local city government. However, they felt having only Missouri and Kansas addresses was too limiting and most city governments would already have access to APIs with address across the nation. They stated, local businesses would most likely need address for the surrounding states. One programmer stated, he might use the site

for teaching purposes, should it remain an open source, but would not find it helpful otherwise due to its limited use.