

Re: A "How would you do this"-type of question. not Java-specific.

Re: A "How would you do this"-type of question. not Java-specific.

Source: <http://coding.derkeiler.com/Archive/Java/comp.lang.java.help/2006-09/msg00499.html>

- *From:* "Jeff" <jeffrey.summers@xxxxxxxx>
 - *Date:* 27 Sep 2006 20:03:01 -0700
-

Eric Sosman wrote:

korto.wow@xxxxxxxx wrote:

Imagine you are writing a program that will display a map of the United States. Your program will ask a user to input their ZIP code. Once they do, a dot will appear on the map to indicate where that ZIP code is located.

My question is, how would you go about doing something like this? Would you have a file of all ZIP codes and their longitude, latitude coords and extrapolate those to x,y coords on your graphic map? Or, is there an easier way?

I can't think of a better approach. The assignment of ZIP codes (USA postal codes) to geographical location is an arbitrary encoding -- some large-scale patterns are evident but they're not regular enough that you could derive map coordinates from numerical calculations on the codes.[*] May as well treat them as arbitrary keys and look them up in a table.

Extrapolation doesn't seem to enter the picture, though.

[*] Mathematically, of course, one could always build a polynomial of degree 100000 or less that interpolates every five-digit ZIP code to its latitude, and another polynomial for longitude. You might, however, have some difficulty in evaluating such high-degree polynomials with useful accuracy! In fact, I rather suspect that simply storing the coefficients (as pairs of `BigInteger`s representing rational numbers, say) might take more memory than the average 32-bit JVM can supply.

Eric Sosman

Re: A "How would you do this"-type of question. not Java-specific.

Re: A "How would you do this"-type of question. not Java-specific.

esosman@xxxxxxxxxxxxxxxxxxxx

Also, you probably don't need all 100,000 entries. For a map, you could probably just look at the first 3 or at most 4 digits to get a dot of reasonable size placed on the map.

.