Client-side geo-processing using WebAssembly and Visual Programming.

name	Jos Feenstra
nr	4465768
email	feenstrajos@gmail.com
date	15-09-2021

Within the domain of Geomatics, geodata experts often would like to take some geo-dataset, and process it for a specific use case. This "geoprocessing" is most often done natively on a desktop by using a library like *CGAL* or a tool like *QGis*. On the web, "Geoprocessing Services" exist like *Google Earth Engine*, which offer server-side geoprocessing.

The aim of this research will be to explore a new geoprocessing method: geoprocessing in a browser, client-side. This would be very beneficial for several reasons. For one thing, it would make geoprocessing tools extremely accessible. Users would not have to install anything besides a web browser, and geodata experts can share tooling without having to download or build anything. Secondly, client-side geoprocessing can make geoprocessing more interactive and insightful. A 'sandbox environment' which can do geodata *retrieval*, *processing*, and *visualization* in a web browser would be a great tool for debugging, quickly looking at the effects of parameters, finding out which algorithms work best with which dataset, etc.

This research will explore the possibilities and limitations of client-side geoprocessing by documenting the creation of such an environment, and putting it to the test. With this environment, this research attempts to solve two big barriers preventing successful client-side geoprocessing. The first barrier is that the client-side programming language `javascript`, together with its library ecosystem, do not offer the speed nor the tools to perform fast geoprocessing.
To solve this, WebAssembly will be considered. WebAssembly is a type of binary that runs in web browsers. It can be used to take an existing C++ geoprocessing library (like cgal), and to publish it in a way anyone with a browser can run it at near native speed.

The second barrier is that an application 'just webassembly' will not be useful enough to make client-side geoprocessing a good alternative to regular geoprocessing. Client-side geoprocessing will eventually require some sort of framework to work within, just like many geoprocessing tools eventually become QGIS plugins. This thesis chooses the framework of a web-based Visual Programming Language, or VPL. the research aims to learn from geo-vpl tools like Save Software's FME, McNeel's Grasshopper and Ravi Peter's GeoFlow in order to strike a balance between user friendliness and, lets say, 'computational control'.

Supervisors: Ken Arroyo Ohori + Stelios Vitalis