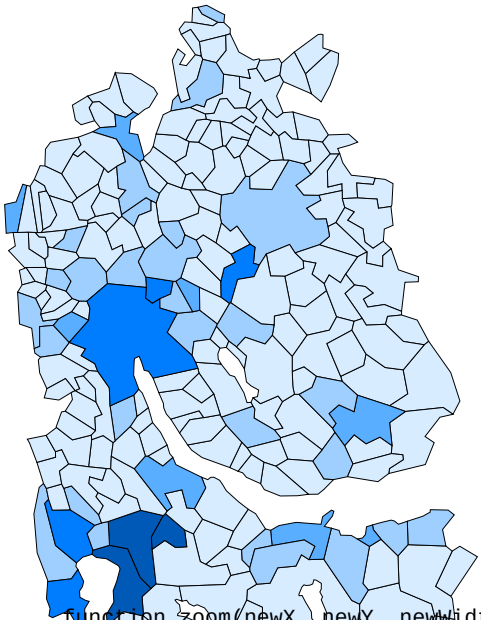
Thematische Karte mit SVG

Vertiefungsblock K7: Kartographie



```
function zoom(newX, newY, newWidth, newHeigth)
{
   var svgdoc;
   if (document.all)
   {
      svgdoc = svgFile1.getSVGDocument();
   }
   else
   {
      svgdoc = document.svgplugin.document.embeds[0].getSVGDocument();
   }
   newViewport = newX + " " + newY + " " + newWidth + " " + newHeight;
   svgMainViewport = svgdoc.getElementById("perimeter");
   svgMainViewport.setAttribute("viewBox", newViewport);
}
```

Raffael Hilber, Stefan Ziegler

9. Februar 2001

Institut für Kartographie
ETH Zürich

WS 2000/2001

Vertiefungsblock K7: Kartographie

Thematische Karte mit SVG

Raffael Hilber, Stefan Ziegler

9. Februar 2001

Vorwort

Diese Arbeit entstand im Rahmen des Vertiefungsblockes K7 “Kartographie”. Der Bericht soll zum Verständnis unserer Arbeit beitragen. Bei allfälligen Fragen oder Anregungen stehen wir gerne zur Verfügung.

Wir möchten allen Personen, die uns im vergangenen Semester betreut und unterstützt haben, unseren herzlichsten Dank aussprechen:

- Prof. Dr. L. Hurni für die Durchführung und Leitung des Vertiefungsblockes K7 WS 2000/2001
- Andreas Neumann für die sehr gute fachliche sowie geduldige Betreuung und Unterstützung
- allen weiteren Personen, die uns in irgendeiner Weise unterstützten

Raffael Hilber
<rhilber@student.ethz.ch>

Stefan Ziegler
<stifu@guv.ethz.ch>

Zusammenfassung

Mit SVG (Scalable Vector Graphics), einem neuen Vektordatenstandard fürs Internet, soll eine interaktive thematische Karte erstellt werden. SVG ist ein offenes, objektorientiertes Datenformat zur Beschreibung von zweidimensionalen Grafiken in XML, das vom World Wide Web Consortium (W3C) empfohlen und zusammen mit diversen grossen Softwarefirmen entwickelt wird. In Verbindung mit einer Programmiersprache – in unserem Fall JavaScript – können diverse Interaktionen und Animationen programmiert werden.

Es wurde ein Prototyp zur Visualisierung der Steuersituation in den Kantonen Schwyz, Zug und Zürich entwickelt, mit dem Ziel, Erfahrungen mit diesem noch in der Entwicklung stehenden Datenstandard zu sammeln.

Das Resultat ist eine Webpage mit Hilfe derer unter anderem die Steuerbelastung auf Gemeinde-, Bezirks- und Kantonsebene für natürliche Personen verschiedener Einkommensklassen verglichen, nach diversen Suchkriterien Werte oder Gebietsnamen gesucht oder der Anteil der juristischen Personen am Steuerertrag verglichen werden kann. Die Homepage ist auf dem Server des Institutes für Kartographie der ETH Zürich in der Rubrik Studentenarbeiten zu finden.

Inhaltsverzeichnis

1 Einleitung

Mit der fortschreitenden Entwicklung der Darstellungs- und Datenverarbeitungstechniken im Internet wird auch dessen Nutzung für die Kartographie attraktiver. Denn lange Zeit fehlte ein geeignetes und effizientes Vektordatenformat, das Interaktionen und Analysen sowie "zooming" (vergrössern und verkleinern) oder "panning" (verschieben) eines gewünschten Bildausschnittes ermöglichte. Der grosse Speicherplatzbedarf und die damit verbundene lange Ladezeit von Rastergrafiken sowie der grosse Programmieraufwand, um einfache Interaktionen zu ermöglichen, beschränkten deren Anwendungsmöglichkeiten für die Kartographie im Internet. Mit der Entwicklung neuer Vektordatenformate wurden diese Probleme immer besser gelöst, so dass mittlerweile diverse Ansätze kartographisch nutzbarer Vektordatenformate existieren. Nennenswert sind Formate mit grosser Verbreitung wie `Flash` von Macromedia (das zudem ein hohes Interaktionsniveau aufweist), `PDF` sowie `ActiveX` oder Formate mit grossen Interaktionsmöglichkeiten wie `SVG` oder `Java2D`.

Alle genannten Formate haben ihre spezifischen Einsatzgebiete und daher auch ihre spezifischen Vor- und Nachteile. Für die Kartographie sind vor allem Formate mit einem hohen Interaktionsniveau interessant. Auf diesem Gebiet konkurrieren sich in erster Linie `Flash` und `SVG` (Scalable Vector Graphics). Vorteile von `SVG` gegenüber Macromedia `Flash` sind, dass `SVG` keine Software sondern lediglich ein Datenstandard ist und daher keine spezielle Software zur Erstellung von `SVG`-Elementen notwendig ist sowie dass das In- und Exportieren von Daten viel einfacher ist. `SVG` ist aber im Moment noch äusserst schwach verbreitet, was aber nicht verwunderlich ist, da dieser Standard noch in der Entwicklungsphase steckt.

1.1 Auftrag

Es soll eine interaktive Karte mit `SVG`, einem neuen Vektordatenstandard fürs Internet, erstellt werden. Das Ziel ist, Erfahrungen mit diesem Datenstandard zu sammeln und aus bestehenden GIS-Datensätzen einen Prototyp einer interaktiven thematischen Karte zu erstellen.

1.2 Thematik

Die Thematik ist in der Aufgabenstellung nicht vorgegeben, konnte daher selber gewählt werden.

Als Datengrundlage standen sowohl für die Thematik als auch für die Geometrie dieselben Daten zur Verfügung, die auch für den Atlas der

Schweiz verwendet wurden. Aus diesem Datensatz galt es daher ein interessantes Thema auszuwählen.

Ein wichtiges Kriterium für die Auswahl des Themas war, dass die Daten möglichst vollständig erhoben sein sollen. Die aktuelle Diskussion über eine mögliche Steuerharmonisierung in der gesamten Schweiz und die Tatsache, dass diese Daten grösstenteils vorhanden sind, gab schlussendlich den Anstoss, die Steuersituation als Karte darzustellen. Das Ziel war die Steuerbelastung für natürliche Personen und den Anteil der juristischen Personen am Steuerertrag zu behandeln. Bei den natürlichen Personen werden vier Gruppen (ledige Personen, verheiratete Personen ohne Kinder, verheiratete Personen mit zwei Kindern sowie Rentner) unterschieden, die jeweils wieder in vier Klassen (Jahreseinkommen bis 30'000 SFr., bis 60'000 SFr., bis 100'000 SFr. und bis 200'000 SFr.) unterteilt werden.

Aus rechtlichen Gründen konnte keine thematische Karte der gesamten Schweiz erstellt werden, denn sowohl die Vektor- als auch die thematischen Daten können sehr einfach aus der Webpage herausgelesen werden. Zudem wären die Vektordaten aller Gemeinden der Schweiz fast 500 Kilobyte gross, was zu Problemen bei der Nutzung übers Internet führen könnte. Deshalb war ein aus etwa drei bis vier möglichst unterschiedlichen Kantonen bestehendes Gebiet gesucht. Die Wahl fiel relativ schnell und eindeutig auf die drei Kantone Zug, Schwyz und Zürich, denn obwohl sie direkt beieinander liegen ist die steuerliche Situation dieser Kantone äusserst unterschiedlich.

Zug ist europa-, wenn nicht sogar weltweit, bekannt als steuergünstiger Kanton. Erwartungsgemäss sind die Steuereinnahmen in den Zuger Gemeinden hoch und stammen vor allem von den zum Teil aktiv in die Gemeinden gelockten juristischen Personen. Grosse Unterschiede in der Steuerpolitik zwischen den elf Zuger Gemeinden sind nicht auszumachen.

Im Kanton Schwyz werden immer stärkere regionale Unterschiede erkennbar. Die Richtung Zürich orientierten ausserschwyzer Gemeinden wie Freienbach oder Wollerau beginnen ihre Standortvorteile (die Nähe zu Zürich und dem Flughafen, den Seeanstoss und die ruhige Wohnlage) voll auszuspielen und können dank der ständigen Zuwanderung von starken Steuerzahlern fast jährlich die Steuerfüsse senken. Dies ganz im Gegenteil zu den innerschwyzer Gemeinden, die zum Teil noch extrem ländlich sind, wie zum Beispiel das Muotathal, wo der Anteil an juristischen Personen verständlicherweise relativ niedrig und die Steuerbelastung ziemlich hoch ist. Im Gegensatz zu Zug und Zürich ist das Steuersystem in Schwyz nicht sehr sozial, es ist nur eine schwache Regression für Personen mit tieferem Einkommen erkennbar. Das heisst, Personen mit niedrigerem Einkommen

erhalten fast keine steuerliche Erleichterungen.

Der Kanton Zürich ist vor allem wegen seiner grossen Städte Zürich und Winterthur mit dementsprechend hohen Steuereinnahmen sowie wegen den vielen extrem kleinen Gemeinden am Rande des Kantons interessant. Überraschend ist, dass es im als sehr urban geltenden Kanton Zürich neben den Städten und den Gemeinden der "Goldküste" auch sehr viele kleine, ländliche und finanzschwache Gemeinden gibt.

1.3 Ziele

Das Ziel dieser Arbeit besteht darin, einen Prototyp einer thematischen Karte mit einem neuen Datenstandard zu erstellen. Das heisst, dass nicht in erster Linie eine perfekt funktionierende und konzipierte Webpage programmiert werden soll, sondern dass das Ziel vielmehr darin besteht, Erfahrungen mit `SVG` zu sammeln und die Programmiersprache `JavaScript` zu lernen.

Bezüglich der zu implementierenden Funktionen stecken wir uns folgende Ziele:

- Vergleich der Steuerbelastung auf Gemeinde-, Bezirks- und Kantonsebene für natürliche Personen. Es soll zwischen verschiedenen Einkommensklassen sowie zwischen ledigen Personen, verheirateten Personen und Rentnern unterschieden werden können. Der betreffende Wert sowie der Gemeinde- und Kantonsname sollen, wenn die Maus über das Gebiet bewegt wird, automatisch angezeigt werden.

- Der Benutzer soll zwischen zwei Arten wählen können, wie die Werte in die Klassen eingeteilt werden; entweder in gleichmässigen Abständen oder in Quantile unterteilt. Zudem soll der Benutzer entscheiden können wieviele Klassen dargestellt werden sollen (zwei bis fünf).

- Vergleich des Anteils der juristischen Personen am gesamten Steuerertrag auf Gemeinde-, Bezirks- und Kantonsebene mittels Kuchendiagrammen. Zudem soll wiederum der Wert sowie der Kantons- und Gemeindenamen "onmouseover" angezeigt werden.

- Anzeigen aller verfügbaren thematischen Daten über eine Gemeinde, einen Bezirk oder einen Kanton in einem neuen Fenster, falls auf das betreffende Gebiet geklickt wird.

- Suchen nach einer Gemeinde, einem Bezirk oder Kanton über die Eingabe des Namens.

- Suchen nach Gemeinden, Bezirken oder Kantonen, welche selbst ausgewählte Kriterien erfüllen. Zur Auswahl stehen alle bei Punkt eins genannten Kriterien.

Ein weiterer Schwerpunkt des Vertiefungsblockes liegt in der Teamarbeit, die bei Programmieraufgaben besonders schwierig ist, da Doppelspurigkeiten vermieden werden müssen und die einzelnen Programmteile jeweils mit möglichst wenig Aufwand wieder ein funktionierendes Ganzes ergeben müssen.

1.4 Verfügbares Equipment und Daten

Verwendet wurden dieselben Vektor- und thematischen Daten, die auch für den Atlas der Schweiz verwendet wurden. Das Copyright dafür liegt beim Bundesamt für Statistik[1] (BfS). Für die Erstellung sowohl der `SVG`-Elemente als auch der `JavaScript`-Funktionen wird keine spezielle Software benötigt, denn sie können in einem normalen Texteditor erstellt werden. Allerdings muss ein Plugin[2] installiert werden, um `SVG`-Dateien in einem Browser betrachten zu können. Zur Bearbeitung der Semesterarbeit standen die Computer im Praktikumsraum des IKA zur Verfügung.

1.5 Geforderte Resultate

Die geforderten Arbeiten sollen im vorgesehenen Zeitrahmen von zwölf Semesterwochenstunden erledigt werden. Nachstehende Arbeiten müssen am Ende des Semesters erledigt sein und dienen so zugleich einem Leistungsnachweis.

- Webpage
- Technischer Bericht
- Plakat
- Präsentation vor Mitstudentinnen und Mitstudenten und den Betreuern

[1] <http://www.statistik.admin.ch>
[2] <http://www.adobe.com/svg/>

1.6 Terminplanung

Für eine Arbeit dieses Umfangs ist es notwendig eine vollständige Terminplanung, welche sich über die ganze Projektzeit erstreckt, aufzustellen. Nachdem der Auftrag analysiert worden ist, können die einzelnen Arbeiten aufgelistet werden. Vorhandene Abhängigkeiten müssen aufgedeckt werden, so dass die Arbeiten in die richtige Reihenfolge eingeordnet werden können. Die Arbeiten werden bezüglich ihres Zeitaufwandes und den benötigten Grundlagen analysiert. Dann kann ein erster provisorischer Zeitplan erstellt werden.

Für die Bearbeitung des Themas sind laut Studienplan zwölf Wochenstunden während des ganzen Semesters vorgesehen. Zur Verfügung stehen der ganze Donnerstag sowie der Freitagmorgen.

Nachfolgend ist eine grobe Reihenfolge der zu erledigenden Arbeiten aufgelistet.

- Wahl der Thematik
- Sichten der Daten
- Erste grobe Definition des Layouts
- Datenaufbereitung mit Hilfe von `Perl`-Skripten
- Einarbeitung in `SVG`
- Erstellen der benötigten `SVG`-Elemente
- Programmierung der dynamischen Elemente mittels `JavaScript`
- Definitives Layout
- Erstellung der Präsentation
- Erstellung des Berichtes
- Erstellung des Plakates

Die Planung eines Prototyps ist meistens schwierig. Vor allem die Erstellung der `SVG`-Elemente sowie das Programmieren waren zeitlich sehr aufwändig, aber schwierig zu planen. Denn viele Ideen für neue Funktionen entstanden während des Programmierens selber. Das wichtigste während dieser Phase war deshalb, nicht zu viel Zeit für fast unlösbare Programmieraufgaben zu verwenden, sondern in kleinen Schritten stetig vorwärts zu kommen.

2 Programmtechnische Grundlagen

2.1 SVG

SVG (Scalable Vector Graphics) ist ein offenes, objektorientiertes Datenformat zur Beschreibung von zwei-dimensionalen Grafiken in XML. Die Sprache wird vom World Wide Web Consortium (W3C) empfohlen und zusammen mit wichtigen Softwarefirmen wie Adobe, die das Plugin für die Browser anbietet, entwickelt. Es können drei Typen von graphischen Objekten angezeigt werden: Vektorgrafiken, Rasterbilder und Text. Die Objekte können gruppiert oder transformiert und deren Eigenschaften, wie zum Beispiel die Farben, verändert werden. Es sind zwei Arten möglich, Grafiken zu animieren oder interaktiv zu verändern: Einerseits können via eine Programmiersprache wie JavaScript Manipulationen am Element vorgenommen werden, andererseits erlaubt SVG gewisse Effekte direkt im SVG-Dokument zu beschreiben. Weil in einem SVG-Dokument sowohl die Grafik als auch die Schrift definiert werden kann, werden unerwünschte Unterschiede in der Darstellung der Schriften auf verschiedenen Browsern vermieden. Mit SVG kann ein beliebiges Koordinatensystem definiert werden, dies ist für kartographische Anwendungen sehr wichtig. Der Koordinaten-Nullpunkt liegt in der linken oberen Ecke, was für die Kartographie nicht unbedingt geeignet ist. Dies kann aber durch einfache Umformungen der Daten korrigiert werden.

Bei geeigneter Programmierung erlauben SVG-basierte Projekte auch eine Offline-Betrachtung und können zum Beispiel auf CD-ROM angeboten werden. In Kombination mit der hohen Qualität, die auch beim Ausdruck erhalten bleibt, ergibt sich so ein reduzierter Arbeitsaufwand, da mit einem Format mehrere Medien angesprochen werden können.

2.2 Perl

2.2.1 Allgemeines

Perl steht für *Practical Extraction And Report Language*. Es ist eine Interpreter-Programmiersprache. Das heisst, ein Programm (besser ein Skript) wird nicht nur einmal – wie bei Compiler-Programmiersprachen üblich – in ein für den Rechner verständlichen Maschinencode übersetzt, sondern jedes Mal wenn eines dieser Skripte aufgerufen wird.

Der grösste Vorteil von Perl ist seine *Portierbarkeit*. Solange es für das jeweilige Betriebssystem einen Perl-Interpreter gibt, läuft das Skript ohne Probleme auf diesem System. Einschränkungen müssen gemacht werden,

falls systemeigene Befehle in das Skript eingebaut werden, so zum Beispiel `wc`, ein beliebtes Tool auf Unixsystemen, das Wörter/Linien von Dateien zählt.

Der grösste Nachteil von `Perl` und Interpreter-Sprachen im Allgemeinen ist die Ausführungsgeschwindigkeit der Skripte, die bei weitem nicht so hoch ist, wie bei compilierten Programmen.

2.2.2 Anwendungsbereich

`Perl` hat heutzutage viele Einsatzgebiete. Einer der ganz grossen Anwendungsbereiche ist die CGI-Programmierung (zum Beispiel für Webshops). Wobei hier `php`, ein ernstzunehmender Konkurrent für `Perl`, immer mehr an Bedeutung gewinnt.

Ein weiteres grosses Einsatzgebiet von `Perl` ist das Umformatieren von ASCII-Dateien. Im Gegensatz zu anderen Programmiersprachen ist das "File-Handling" vereinfacht. Somit wird es leichter Datensätze in ein gewünschtes Format zu bringen.

Dies machen auch wir uns zunutze. So können – auch mit geringen Programmiererfahrungen mit `Perl` – die Datensätze des Bundesamtes für Statistik in eine `SVG`-Datei umgewandelt werden.

2.3 JavaScript

2.3.1 Allgemeines

Dokumente im WWW sind meist mit `HTML` geschrieben. Diese Markup-Language ist leicht zu erlernen und fand auch aus diesem Grund sehr schnell Verbreitung. Internetseiten, die mit `HTML` erstellt werden, sind jedoch statisch. Das heisst, es bewegt sich nichts (genaugenommen gibt es den äusserst unbeliebten `<blink>`-Tag) und der Anwender kann kaum Einfluss auf die Seite ausüben. Mit `JavaScript`, das ebenfalls leicht zu erlernen ist, wird es möglich dynamische Webseiten zu programmieren. Ein weiterer grosser Vorteil von `JavaScript` ist dessen Plattformunabhängigkeit.

Spricht man von `JavaScript` im Allgemeinen meint man eigentlich `ECMAScript`. Dies ist der von der europäischen Organisation ECMA verabschiedete Standard. Eine Standardisierung wurde notwendig, weil sich die beiden *JavaScript*-Versionen von Netscape und Microsoft mehr und mehr unterschieden. Zwar sind sich die beiden Sprachen in weiten Teilen sehr ähnlich, trotzdem gibt es vor allem in der Schnittstelle zu `SVG` grosse Unterschiede, die uns in der Umsetzung einige Probleme bereiteten (siehe Kapitel 4.1.3).

2.3.2 Anwendungsbereich

Wie oben schon erwähnt, findet `JavaScript` heute rege Anwendung, um Webseiten interaktiv zu gestalten. Der Endanwender soll die Möglichkeit bekommen, auf die Seite Einfluss zu nehmen und sie auch verändern können.

Auch wir benutzen `JavaScript` um die eigentliche Interaktivität der Webkarte zu programmieren. Die vorhandene Schnittstelle zu `SVG` ermöglicht es aber erst die gewünschten Interaktivitäten an der Karte selbst vorzunehmen.

3 Programmtechnische Umsetzung

In diesem Kapitel werden die notwendigen Schritte, die zum Endprodukt führen, erläutert. Beginnend bei der Konvertierung der Daten mit `Perl`-Skripten wird später auf die eigentliche Programmierung der Karte eingegangen und die möglichen Funktionen bzw. die Interaktionsmöglichkeiten kurz erklärt.

3.1 Konvertierung der Daten

Sämtliche Daten (Geometrie *und* Thematik) werden mit selbstgeschriebenen `Perl`-Skripten in eine für `JavaScript` bzw. `SVG` kompatible Form gebracht. Die thematischen Daten werden mit Arrays direkt in `JavaScript` integriert.

Mit den geometrischen Daten wird ein erstes `SVG`-File erzeugt, das eigentliche Grundgerüst der Karte. Später muss gegebenfalls das File ergänzt werden. Auch dies geschieht mittels dem erweiterten `Perl`-Skript.

Die thematischen Daten werden direkt in `JavaScript`-Variablen konvertiert. Es handelt sich dabei um Arrays, welche neben den eigentlichen Daten auch Informationen über die Bezirks- und Kantonszugehörigkeit enthalten können.

3.2 Programmaufbau

3.2.1 Allgemeines

Die Interaktionen, welche unsere Karte enthalten soll, werden alle mit `JavaScript` programmiert. Die thematischen Daten sind in globalen Arrays gespeichert, weil der Zugriff immer gewährleistet sein muss. Die Funktionen sind in zwei separaten Files geschrieben (wobei das erste zudem die thematischen Daten enthält). Dies erhöht die Lesbarkeit bei etwa 3'000 Zeilen Quellcode.

3.2.2 Verwaltungsebene wählen

Wählt man eine der drei möglichen Verwaltungsebenen – Gemeinde, Bezirk oder Kanton – aus, wird die Funktion `showGrenzen` aufgerufen. Die gewünschte Ebene wird in der Karte sichtbar, die anderen zwei unsichtbar. Zudem wird, je nach gewünschten Thema, die Funktion `colorit` oder `coloritJur` aufgerufen. Diese beiden Funktionen werden weiter unten im Kapitel 3.2.3 näher beschrieben.

3.2.3 Thema wählen

Das gewünschte Thema lässt sich aus einem Formular auswählen. Zur Verfügung stehen (a) *Steuerbelastung für verschiedenen Einkommensklassen*, (b) *Anteil der juristischen Personen am totalen Steuerertrag* und (c) *Suche nach Namen oder Daten*.

Bei (a) wird die Funktion `coloritJur` aufgerufen, welche die Verwaltungseinheiten einfärbt. Zudem ist sie verantwortlich für die Einfärbung und Beschriftung der Legenden.

Wird (b) gewählt, erscheint ein neues Formular, aus dem die gewünschte Einkommensklasse gewählt werden kann. Dann wird die Funktion `colorit` aufgerufen, die sehr ähnlich zur Funktion `coloritJur` ist. Für (c) siehe Kapitel 3.2.5.

Die beiden Funktionen lesen aus den beiden "Statistik"-Formularen, welche Anzahl Klassen und welche Klassifikation (Quantile oder gleiche Intervalle) gewählt wurde. Ändert man eines dieser Formulare, rufen sie ihrerseits wiederum die passende `colorit`-Funktion auf.

3.2.4 OnMouseOver-/OnClick-Effekte

Bewegt man mit den Mauszeiger über die Karte, werden einige Funktionen ausgelöst. In einem `SVG`-File erscheint der Name des Gebietes, die Kantonszugehörigkeit sowie der entsprechende Wert. In einem weiteren `SVG`-File werden Kuchendiagramme gezeichnet, welche die relativen und absoluten Werte der *Anteile der juristischen Personen am totalen Steuerertrag* angeben.

Klickt man auf eines der Gebiete erscheint ein neues Fenster, das alle Daten des Gebietes auflistet.

3.2.5 Suchfunktionen

Es wurden einige interessante Suchfunktionen implementiert. Zuerst muss man sich entscheiden auf welcher Verwaltungsebene man suchen will, wobei aber jeweils alle Ebenen sichtbar gemacht werden können. Dann entscheidet man sich für eine Suche nach Namen oder nach Daten. Bei einer Namenssuche ist als Eingabe auch nur ein Teil des Namens gültig, weil mittels eines regulären Ausdrucks gesucht wird. Für die Datensuche stehen alle vorhandenen Daten zur Verfügung. Die Suchergebnisse lassen sich anschliessend in der Karte anzeigen. Zudem kann eine erfolgreiche Suche mit neuen Suchkriterien weiter eingeschränkt werden.

Die Suchalgorithmen wurden alle selbst geschrieben. Somit sind sie

nicht so mächtig wie Abfragen mit `SQL`, brauchen jedoch auch keine Anbindung an eine Datenbank .

4 Diskussion

4.1 Probleme

Natürlich läuft bei einer Arbeit dieses Umfanges nicht alles reibungslos ab. Es ist klar, dass es auch Probleme gibt. In diesem Kapitel sollen daher aufgetretene Probleme und in einem nächsten Schritt Lösungsansätze dazu beschrieben werden.

4.1.1 Geometriedatensatz

Problem

Wie schon in Kapitel 1.4 erwähnt, stammen sowohl die geometrischen als auch die thematischen Daten vom Bundesamt für Statistik. Mit den geometrischen Daten gab es Probleme, falls im jeweiligen Gebiet ein See zu liegen kam. Dieser See wurde dann einfach durch eine geraden Strich mit einem Grenzpunkt des Gebietes verbunden.

Lösung

Dieses Problem konnte einfach, jedoch zeitaufwändig, durch Handarbeit behoben werden. Falls das betrachtete Gebiet zu gross wäre, um dieses Problem zu lösen, käme zum Beispiel folgende Lösung in Frage: Die Verbindung eines Seepunktes mit einem Grenzpunkt ist meistens viel länger als ein normaler Grenzabschnitt. Es könnte daher einfach nach den längsten Strecken gesucht werden.

4.1.2 Formatierung von Zahlen

Problem

In `JavaScript` gibt es keine Funktion oder Methode, die es ermöglicht Zahlen in einem vom Programmierer gewünschten Format auszugeben. Gewünscht wären Formatierungen wie das Runden auf eine bestimmte Anzahl Nachkommastellen (`5.568456987` wird zu `5.57`) oder das Einfügen von Tausender-Hochkommata (`807523489` wird zu `807'523'489`).

Lösung

Für Letzteres musste extra eine eigene Funktion geschrieben werden. Man übergibt ihr die Zahl und der Rückgabewert ist wiederum die Zahl, jetzt

mit den Hochkommata. Jedoch ist man nicht auf die Hochkommata beschränkt, wählbar ist jedes beliebige Zeichen. Es wird empfohlen die Funktion gerade erst *vor* der Ausgabe auf dem Bildschirm aufzurufen, denn die Zahl wird durch diese Manipulationen in einen String umgewandelt.

Das Runden der Nachkommastellen erweist sich als schwieriger. Gewählt wurde eine nicht ganz korrekte Methode. Die Zahl wird zuerst in einen String umgewandelt (`+" "`). Dann wird der String mit der `substr()`-Methode vom ersten Zeichen an bis zur gewünschten Stelle extrahiert. Diese ziemlich unelegante Methode hat zwei unschöne Nebenerscheinungen: Einerseits wird nicht im mathematischen Sinne gerundet und andererseits haben Zahlen mit einer ungleichen Anzahl Stellen vor dem Komma nicht gleich viele Nachkommastellen.

Für eine elegantere Lösung wäre ein mathematischer Ansatz denkbar: Die zu rundende Zahl wird mit einem Faktor 10^x multipliziert, dann entweder auf- (`ceil()`) oder abgerundet (`floor()`), und anschliessend wieder durch 10^x dividiert.

4.1.3 Unterschiede zwischen MS Internet Explorer und Netscape Navigator

Problem

Die Karte kann nur auf dem Microsoft Internet Explorer betrachtet werden. Die Objekthierarchien (DOM) sind im Internet Explorer und Navigator verschieden aufgebaut. Darum müssen die `SVG`-Objekte auch immer auf zwei Arten angesprochen werden. Anfängliche Probleme bei der Definition der `CSS` für den Netscape Navigator veranlassten uns leider die Version für den Navigator zu Gunsten der Programmierung neuer Funktionalitäten zu vernachlässigen.

Lösung

Mit einem relativ geringen Zeitaufwand kann die Webpage Netscape-kompatibel gemacht werden.

4.1.4 Konzeption der Arbeit

Problem

Uns war es nicht möglich gleich zu Beginn eine ganz klare Konzeption der Arbeit aufzustellen. Dies hängt damit zusammen, dass wir unser Thema relativ frei wählen konnten und dass das Programmieren immer auch ein

bisschen nach dem Prinzip *Try And Error* abläuft. Befindet man sich in einer Sackgasse, muss man zurückgehen und einen anderen Weg wählen. Zudem entstanden einige der besten Ideen erst im Verlauf der Arbeit.

Lösung

Für dieses Problem kann keine "08/15-Lösung" angegeben werden. Das obengenannte Problem ist in einem gewissen Teil so gar nicht lösbar und für andere Arbeiten wird es wieder anders aussehen.

4.1.5 Teamarbeit

Problem

Teamarbeit ist während einer solchen Arbeit sicher wichtig und auch sinnvoll. Jedoch muss die Teamarbeit sehr gut koordiniert sein. Es gab vor allem Probleme, weil wir eigentlich meist an verschiedenen Funktionen arbeiteten. Das Zusammensetzen dieser einzelnen Programmteile erwies sich immer als ein langwieriger und mühsamer Prozess.

Lösung

Auch hier kann keine allgemeingültige Lösung angegeben werden. Sinnvoll sind jedoch möglichst wenig globale Variablen (wobei das wohl grundsätzlich fürs Programmieren gilt), so dass die einzelnen Funktionen autonom werden und so einfach zusammengefügt werden können.

4.1.6 Zoom-Tool

Problem

Das selbst entwickeltes Zoom-Tool hat einige Mängel. Es sind nur die Zoom-Stufen 1-, 4- oder 16-fach möglich. Und falls man sich dann mit Hilfe der Pfeile in der Karte bewegt, ist in der Zoom-Übersichtskarte nicht ersichtlich, wo man sich gerade befindet.

Lösung

Durch geometrische Randbedingungen sind die Zoom-Stufen vorgegeben und können somit nicht geändert werden. Eine weitere Vergrösserung wäre zwar möglich, aber kaum sinnvoll ($4^3 = 64$). Um eine bessere Übersicht

von seinem Standpunkt zu erlangen, müsste zum Beispiel ein farbiges Viereck auf der kleinen Karte vorhanden sein, das jeweils die Bewegungen des Anwenders mitmacht.

4.1.7 Aggregierung der thematischen Daten

Problem

Die Daten der Steuerbelastung und diejenigen des Steuerertrags sind immer noch getrennt in zwei verschiedenen "Array-Gruppen" gespeichert. Dies weil es am Anfang für das Programmieren einfacher schien. Später hat sich aber gezeigt, dass dadurch der Programmieraufwand grösser wurde, da für diverse Funktionen Werte aus beiden Tabellen gebraucht wurden.

Lösung

Ein Zusammenführen hätte zur Folge, dass viele Funktionen, die bereits einwandfrei funktionieren, wieder hätten umprogrammiert werden müssen. Dies kann nur mit einem beträchtlichen Mehraufwand an Zeit erledigt werden.

4.1.8 Statistische Eckdaten

Problem

Im Fenster, das alle Suchfunktionen enthält, fehlen statistische Eckdaten, damit ein unerfahrener Benutzer weiss, in welchem Bereich die Zahl, die er als Suchkriterum eingibt, liegen muss, damit die Suche einigermassen sinnvoll ist.

Lösung

In einem zusätzlichen `<div>`-Layer können die gewünschten Eckdaten ein- und ausgeblendet werden.

4.2 Erweiterungsmöglichkeiten

Die Arbeit könnte sowohl bezüglich der Thematik als auch der implementierten Funktionen erweitert werden.

4.2.1 Thematik

Der reine Vergleich der Steuersituation in den drei Kantonen zeigt einige interessante Details. Der Vergleich könnte aber ausgeweitet werden, denn die Höhe der Steuern sagt noch relativ wenig darüber aus, ob ein Wohnort für eine Person oder Familie finanziell vorteilhaft ist. Um einen etwas umfassenderen Vergleich zwischen den Gemeinden oder Kantonen machen zu können, wäre eine Verbindung mit der jeweiligen Höhe der Wohnungsmieten und der Grundstückspreise interessant.

4.2.2 Funktionen

Print-Button

Die angezeigte Karte inklusive der Legende und dem Titel soll ausgedruckt werden können. Durch Klicken auf einen "Print-Button" wird ein neues Fenster geöffnet, das die drei oben genannten Elemente enthält. Das Format des Ausdrucks, sowie die Farben können vom Benutzer gewählt werden. Das Wählen der Farbe ist wichtig, da eine Farbe auf dem Bildschirm gut aussehen kann, jedoch auf Papier nicht. Zudem gibt es auch Unterschiede zwischen den Druckern selbst.

Andere OnMouseOver-Effekte

Sinnvoll wäre wenn die Informationen über ein Gebiet direkt beim Mauszeiger erscheinen würden, denn oft ist es mühsam den Mauszeiger auf einem Gebiet zu halten und mit den Augen auf einen anderen Teil des Bildschirms zu fokussieren.

Server-Sided SVG

Dies ist die kühnste und zugleich auch am wenigsten ausgereifte Idee. Bis anhin wird die gesamte Geometrie des kompletten Gebietes beim Aufrufen der Webseite einmal ganz geladen. Das kann bei grossen Gebieten (also grossen `SVG`-Files) und einer langsamen Internetverbindung längere Wartezeiten verursachen. Die Idee ist nun, dass der Anwender mittels einer Übersichtskarte dem Server mitteilt, welchen Teil des Gebietes er zu sehen wünscht. Erst dann wird der ausgewählte Teil vom Server verschickt. Die Geometrie kann somit zuerst komplett auf einen Server ausgelagert werden, was sicher zu einer gewissen Geschwindigkeitssteigerung führen kann.

Denkbar ist auch die Auslagerung der thematischen Daten auf einen Server. Durch eine Anbindung an eine Datenbank wären dann komplexere Abfragen möglich.

Sinnvoll sind solche Massnahmen jedoch nur bei sehr grossen Gebieten oder wenn eine grosse Menge an thematischen Daten vorhanden ist.

5 Schlusswort

Der Reiz dieses Vertiefungsblockes bestand darin, mit einem Datenstandard zu arbeiten, der noch in der Entwicklungsphase steckt und dadurch an diesem Beispiel die äusserst dynamische Entwicklung eines neuen Internetstandards etwas mitverfolgen zu können. Ein Nebeneffekt davon ist natürlich, dass dadurch keine Handbücher und nur wenige Tutorials zum Thema existieren. Eine gute und ausführliche Dokumentation zu `SVG` ist auf der Homepage des W3C zu finden, aber die Schnittstelle zwischen `JavaScript` und `SVG` ist nur spärlich dokumentiert. Doch gerade dies hatte auch gewisse Vorteile, denn somit lernten wir immer besser und schneller die nötigen Informationen aus den verschiedensten Quellen zusammenzusuchen.

Trotz anfänglichen Schwierigkeiten und Problemen verloren wir unsere Freude am Erstellen dieser Arbeit nie. Wir sind uns bewusst, dass wir nun keine `SVG`- und `JavaScript`-Experten sind. Dies war aber auch nicht das eigentliche Ziel des Vertiefungsblockes. Vielmehr sollten wir anhand dieses Prototyps das Potential von `SVG` für die Kartographie kennenlernen.

Literatur

[1] ADOBE SYSTEMS INCORPORATED (2001): *SVG Tutorial*, <www.adobe.com/svg/tutorial/intro.html>

[2] GULBINS, Jürgen; KAHRMANN, Christine, (1993): *Mut zur Typographie, Ein Kurs für DTP und Textverarbeitung*, Springer-Verlag, Berlin

[3] HAJJI, Farid (2000): *Perl, Einführung, Anwendungen, Referenz*, Addison-Wesley, München

[4] HURNI, Lorenz; SPIESS, Ernst (1998): *Kartographie Grundzüge (Skript)*, Institut für Kartographie (IKA), ETH Zürich

[5] KOCH, Stefan (1999): *JavaScript, Einführung, Programmierung, Referenz*, dpunkt.verlag, Berlin

[6] NEUBAUER (1996): *Feinheiten bei wissenschaftlichen Publikationen – Mikrotypographie-Regeln, Teil 1*, Die TeXnische Komödie 4/1996, S. 23-40

[7] NEUMANN, Andreas (2000): *Social Patterns of Vienna, Statistical Maps with SVG*, <www.karto.ethz.ch/~aneumann/cartography/vienna>

[8] W3C, World Wide Web Consortium (2000): *Scalable Vector Graphics (SVG) 1.0 Specification, Candidate Recommendation*, 02 November 2000, <www.w3c.org/TR/SVG>

[9] WINTER, Andreas (2000): *Dynamically generated statistical Maps of Europe by using SVG*, <www.carto.net/papers/svg/diagr.html>

Abkürzungen

ASCII	American Standard Code for Information Interchange
CSS	Cascading Stylesheet
DOM	Document Object Model
ETH	Eidgenössische Technische Hochschule
GIS	Geographisches Informationssystem
HTML	Hypertext Markup Language
IKA	Institut für Kartographie
PDF	Portable Document Format
Perl	Practical Extraction And Report Language
SVG	Scalable Vector Graphics
W3C	World Wide Web Consortium
XML	Extensible Markup Language