Praktikum zur Veranstaltung XML-Technologie:

Übung 10

Projekt „Geodaten“

Organisatorisches

- Arbeitsverzeichnis:

  `~/lv/xmltech/10/`

- Dateinamen:

  `10-route.gpx, 10-karte.osmx, 10-geo2svg.xsl,`
  `10-osmx.xsd, 10-karte.svg, 10-karte.css`

- Abzugeben:

  `alle`

- Werkzeuge:

  `(X)Emacs` Nutzen Sie den XML-Modus!

  `Firefox` Zur Kontrolle des SVG-Outputs

  `xalan` Der XSLT-Prozessor (optional)

  Die angegebenen Websites

Erläuterungen: OpenStreetMap, OSM

- Geografische Informationen lassen sich einfach beschreiben mit
  - Punkten, bestehend aus einem Längen- und einem Breitengrad,
  - Wegen (Punktfolgen) und
  - Flächen (geschlossenen Wegen, letzter Punkt = erster Punkt)
- Das Projekt OpenStreetMap (http://www.openstreetmap.org/) erfasst und erzeugt auf diese Weise Landkarten. Jeder kann dazu beitragen durch Einarbeitung selbst erfasster und korrekt attributierter OSM-Daten.
  - Das OSM-Format stellt einen einfachen XML-Dokumenttypen dar.

- Wege und Flächen werden mit „Tags“ präzise charakterisiert. Ein „Weg“ kann z.B. ein Stück Autobahn, ein Fußgängerweg, eine Eisenbahnlinie, aber auch eine Flugroute oder ein Fluss sein. Flächen können Gebäude, Sportplätze, Wiesen, Wälder etc. sein.
- Die beschreibenden Tags bestehen i.w. aus standardisierten „keys“ und dazu passenden „values“. Dokumentation dazu hier:
  - http://wiki.openstreetmap.org/wiki/De:Elemente
  - http://wiki.openstreetmap.org/wiki/De:Map_Features

Erläuterungen: GPS, GPX

- Einige Routenplaner sind in der Lage, geplante Routen im GPX-Format (einem anderen XML-Dokumententypen für Geodaten) zu exportieren. Mobile GPS-Geräte oder „Navis“ können diese standardisierten Daten importieren und z.B. die aktuelle Position zusammen mit der Route anzeigen.
  - GPX wurde von der Fa. Topografix entwickelt und ist frei verwendbar. Es ist wesentlich umfangreicher als für unseren Bedarf erforderlich, z.B. erfasst es auch Höhenangaben.
  - Das Schema gibt es hier: http://www.topografix.com/GPX/1/1/gpx.xsd
- Einige GPS-Geräte erfassen auch zurückgelegte Wege und exportieren sie als GPX-Daten. Daraus lassen sich z.B. OSM-Daten ableiten und auf diese Weise neue Kartenteile generieren.
- Ein freier Routenplaner im Web mit Routenexport per GPX ist z.B.:
  - http://www.radroutenplaner.hessen.de/

Projektziel

- Ziel: Anzeige einer selbst generierten Route im GPX-Format auf einem im OSM-Format exportierten Kartenausschnitt
- Vorgaben
  - **Zielformat ist SVG**
  - Beide Quelldateien sind mit einem XSLT-Skript ins Zielformat zu transformieren. Die Kartenquelle „importiert“ dazu die Routen-Datei
  - Für das OSM-Format ist
    - eine Erweiterung für den Import von GPX-Routen vorzusehen und
    - ein Schema zu entwickeln, mit dem eine Validierung der gewählten Eingabedatei möglichst präzise möglich ist
  - Gestaltungsaspekte, insbesondere Strichdicken und -Farben sowie Farben von Flächen sind per CSS auf der Basis von Klassen zu erzeugen. Abstrakte Tag-Angaben werden dazu in Klassen gewandelt, diese per CSS für sinnvolle Darstellungen verwendet. Als Anregungen können die Kartendarstellungen von OpenStreetMap.org oder maps.google.de dienen.

Projektziel

- Vorgaben (Forts.)
  - Der SVG-Ausschnitt soll variabel einstellbar sein (Default: 800 x 600 px, per Parameter „w_x“ und „w_y“ von außen überschreibbar) und die Quelldaten möglichst unverzerrt darstellen.
  - Dazu soll das XSLT-Skript die Extremwerte aller Punkte (Karten- wie auch Routenpunkte) ermitteln und die Längen- und Breitengrad-Angaben geeignet transformieren.
  - Die Extremwerte „lon_min, lon_max, lat_min, lat_max“ sollen ferner per Parameter vorgebbar sein und dann Vorrang vor den automatisch ermittelten haben.
  - Kartentexte (Straßennamen) sollen in der Karte per SVG-Element „text“ angezeigt werden.

Dateien

- **10-route.gpx**
  a) Selbst messen, z.B. mit einem geeigneten GPS-Gerät mit Exportfunktion auf dem Heimweg
  oder:
  b) Mit dem Radroutenplaner erzeugen und exportieren, z.B. Ihren Weg von der FH nach Hause (falls nah genug)
  oder:
  c) Mit Google Maps Koordinaten einzelner Punkte einer frei gewählten Route ermitteln und daraus per Editor die Datei unter Beachtung des GPX 1.1-Schemas erzeugen

  – Auf jeden Fall: Vor Verwendung schema-validieren (GPX-Version 1.1)!
  – (Die Schemadatei finden Sie z.B. im Verzeichnis „10“)

Dateien

- **10-karte.osmx**
  - Einen zur Route passenden Kartenausschnitt bei OpenStreetMap auswählen und im OSM-Format exportieren
  - Ein guter Detailgrad liegt bei einer Dateigröße von ca. 500 kB vor
  - **Jedes Team soll einen eigenen Ausschnitt wählen!**
  - Die OSM-Datei erweitern um ein selbst definiertes Element (optional: mit eigenem Namensraum), um auf zu importierende Routen-Dateien im GPX-Format verweisen zu können. Als Verweis genügt i.a. die Angabe eines URL.
  - Ebenfalls ist die OSM-Datei um einen Namensraum und um einen Verweis auf eine Schemadatei „10-osmx.xsd“ zu erweitern. Wegen dieser Erweiterungen nennen wir sie nun auch „OSMX“-Datei (x = extension).

Dateien

- **10-osmx.xsd**
  - Diese Schema-Datei ist neu zu entwickeln. Sie muss nicht alle Features von OSM beherrschen, wohl aber die in 10-karte.osmx vorkommenden.
  - Beschreiben Sie die auftretenden Datentypen möglichst präzise. Insbesondere sollte der Datentyp zum Tag-Attribut „k“ nicht einfach „string“ sein, sondern per „enumeration“-Facette gebildet werden. Die hier zulässigen Werte helfen Ihnen auch bei der CSS-Gestaltung.

- **10-geo2svg.xsl**
  - **Das zentrale XSLT-Transformationsskript!**
    Weitere Vorgaben dazu siehe unten.

Dateien

- **10-karte.css**
  - Die CSS-Datei, auf die 10-karte.svg verweist.
  - Sie soll insbesondere die Transformationsergebnisse der verschiedenen Wege- und Flächen-Klassen geeignet darstellen (Wahl von Farbe, Strichbreite etc.)

- **10-karte.svg**
  - Die Ausgabe, zu erzeugen etwa per Kommandozeile:

```
xsltproc 10-geo2svg.xsl 10-karte.osmx > 10-karte.svg
```

Übung 10 (Projekt) – das XSLT-Stylesheet

Überblick – Funktionelle Blöcke / Teilaufgaben:

- Deklaration diverser Namensräume
- Vereinbarung globaler Parameter und „Konstanten“
- Extremwertbestimmung der Längen- und Breitengrade, damit dann Vorbereitung der Koord.-Transformation
- Koordinatentransformation
- Organisation des SVG-Ausgabedokuments
  Dabei Unterscheidung zwischen Pfaden und Flächen
- Einbindung der externen Route

Übung 10 (Projekt) – das XSLT-Stylesheet

Die Koordinatentransformation

- Sei λ ein Längengrad, β ein Breitengrad, P(λ,β) ein durch diese Winkel definierter Punkt auf der Erdoberfläche.
- Wir konstruieren eine Tangentialebene E an P

  (Vereinfachung: Kugel, kein Ellipsoid; Erdradius r ≈ 6371 km)
- Koordinatensystem von E:

  y weist in Richtung Norden, x Richtung Osten
- **Lineare Näherung**, gültig für kleine Winkeländerungen Δλ, Δβ:

$$x(\Delta\lambda, \Delta\beta) = r \sin(\Delta\lambda) \cos(\beta) \approx r * \cos(\beta) * (\pi/180) * \Delta\lambda \quad (\Delta\lambda \text{ in Grad})$$
$$y(\Delta\lambda, \Delta\beta) = r \sin(\Delta\beta) \approx r * (\pi/180) * \Delta\beta \quad (\Delta\beta \text{ in Grad})$$

- Für kleine Kartenausschnitte wie z.B. bei Stadtplänen ist dies eine sehr gute Näherung. Dann gilt auch: cos(β) ≈ const.

Die Koordinatentransformation

- Ausgehend vom x/y-Koordinatensystem können Sie nun leicht in Ihr Bildschirm-Koordinatensystem umrechnen. Sie benötigen dazu die Extremwerte der auftretenden Winkel, (λ_0,β_0) und (λ_1,β_1).
- Achten Sie darauf, dass Ihr Bild maßstabsgerecht ist, d.h. dass in x- und y-Richtung gleiche Strecken gleiche Entfernungen bedeuten. **Verzerren Sie Ihre Karte nicht!**
- Neben einem (konstanten) Skalenfaktor erhalten Sie noch eine Richtungsumkehr in der vertikalen Achse, weil der Ursprung bei SVG oben links liegt und die vertikale SVG-Achse abwärts zeigt.

- Tipps
  - Legen Sie Konstanten global an
  - Ermitteln Sie Parameter zur Umrechnung nur einmal (leider nicht global möglich)
  - Reichen Sie diese zum Transformation-Template durch.

Übung 10 (Projekt) – das XSLT-Stylesheet

Einbindung der externen Route

- Verwenden Sie die XSLT-Funktion „document()“ !
- Ergänzen Sie Schablonenregeln, die die GPX-Elemente verarbeiten
- Oder: Erweitern Sie existierende Schablonenregeln, sodass diese auch „passende“ GPX-Elemente verarbeiten

- Tipps:
  - Beide Dokumenttypen kodieren Längen- und Breitengrad gleichartig, nämlich als Attribute „lon“ und „lat“ eines Elements, das einen Punkt bzw. Knoten repräsentiert. Dies kann Ihnen eine Transformations-schablone ersparen!
  - Die Koordinatensysteme von OSM und GPX sind gleich! Ihre Route muss daher exakt zu Ihrer Karte passen, sonst stimmt etwas nicht.

Übung 10 (Projekt) – das XSLT-Stylesheet

Pfade/Wege bzw. Flächen

- OSM verwendet für beide Objekte dasselbe Element „way“
- Flächen sind erkennbar als geschlossene Wege, d.h. das erste und das letzte Element „nd“ referenzieren denselben Knoten („node“).

- Tipp:
  - Realisieren Sie Wege per „polyline“
  - Realisieren Sie Flächen per „polygon“. Lassen Sie dabei den letzten Wegpunkt aus, weil „polygon“ implizit den Weg schließt.

SVG-Organisation

- Wählen Sie einen konstanten Titel und eine konstante Beschreibung.
- Realisieren Sie die gesamte Karte als ein Symbol, das Sie per „use“ zur Anzeige bringen.

  Eine Ausschnittskorrektur ist dann leicht per Transformation in „use“ durchführbar!

  Bem.: Größenänderungen sind so nicht wirklich befriedigend, weil sich auch Strichbreiten verändern…

- Die leidige Reihenfolge…
  - Abarbeiten in Dokumentenreihenfolge liefert teils unbefriedigende Ergebnisse.

    Beispiel: Die (später gelesene) Parkfläche überdeckt Ihre Parkwege ☹
  - Umsortieren der Inputs kann aufwändig werden und soll hier nicht geschehen. ABER: Stellen Sie sicher, dass zumindest die **Route immer „oben“** liegt!

Übung 10 (Projekt) – Styling & die CSS-Datei

- **Identifikation**
  - Vergeben Sie Attribute „class“ an Ihre Elemente „polyline“ und „polygon“
  - Der Inhalt von „class“ wird bestimmt von „tag“-Elementen im jeweiligen „way“-Element. Nicht jedes „tag“ ist verwertbar.
- Beispiel: <tag k="highway" v="primary"/>

  → class="highway-primary"

- **CSS-seitige Umsetzung**
  - Statt nur zwischen „polyline“ und „polygon“ unterscheiden zu können (oder gar Styles direkt im XSLT-Quelltext zu vergeben), können Sie nun das Aussehen der Karte weitgehend mit der CSS-Datei bestimmen.
  - Beispiel: .highway-primary {fill: none; stroke: yellow; stroke-width: 6px;}
  - Vorgabe für die **Route** (track): Rot, mind. 4px dick!

Übung 10 (Projekt) – Text

- Wir beschränken uns hier auf die Anzeige von **Straßennamen** – keine Flächen/Gebäude, keine Knoten!
  - Straßennamen findet man in Tags von „way“ mit k="name"
  - Beispiel: <tag k="name" v="Kurt-Schumacher-Ring"/>
- Der Einfachheit halber wird jeder Text so angezeigt:
  - Gleich groß (8pt) und mit gleicher Farbe (z.B. dunkelgrau)
  - Horizontal v.l.n.r., beginnend am ersten „node“ des Weges
- CSS-seitige Umsetzung
  - Beispiel: Ausschalten aller Texte, etwa zum Testen

    text {display: none;} /* Die regulären Statements sind Ihre Aufgabe */

Organisatorisches

- Bearbeitung
  - Der Projektumfang ist für **Zweierteams** ausgelegt – bitte arbeitsteilig vorgehen!
  - Vorschlag: Schwerpunkt 1 = XSLT, 2 = CSS + XSD + „Mathe“

- Abgabe:
  - Zu Beginn der letzten Übung, also **am 23. Juni 2009**

- **Zusätzlich Abnahme**:
  - Jedes 2er-Team stellt sein Ergebnis der Gruppe kurz vor bzw. beschreibt, bis wohin es kam und welche Hindernisse sich nicht beseitigen ließen. **Zeit pro Team: <= 10 Min.**
  - Dazu sollen sich die Dateien auf dem Fileserver befinden und vom Dozenten-PC aus anzeigbar sein – bitte ggf. rechtzeitig von privaten PCs kopieren und im Linux-Cluster testen!