

---

# Inhaltsverzeichnis

---

## Teil I Protokolle vom 4.11.2004 bis 18.11.2004 (Patrik Marschalik)

---

### XML Namespaces

*Sebastian Koske, Rafael Grote* ..... 3

### Scalable Vector Graphics (SVG)

*Tinosch Ganjineh, Christian v. Prollius* ..... 5

### Synchronized Multimedia Integration Language (SMIL)

*Georg Sisow, Mattias Hilliges* ..... 9



**Protokolle vom 4.11.2004 bis 18.11.2004  
(Patrik Marschalik)**



---

# XML Namespaces

Sebastian Koske und Rafael Grote

## Vortrag

Der Vortrag gliedert sich in die Abschnitte Motivation, Umsetzung und Definition in XML, Default-Namespaces und Gültigkeitsbereich.

### Motivation

Bei der Benennung von Elementen und Attributen in XML kann es unter Umständen zu Konflikten kommen. Man unterscheidet zwischen semantischen und syntaktischen Namenskonflikten. Ein semantischer Namenskonflikt liegt vor, wenn bei unterschiedlicher Bedeutung der gleiche Namen vorliegt. Von syntaktischem Namenskonflikt spricht man, wenn scheinbar gleiche Elemente eine unterschiedliche Struktur aufweisen.

Um dieses Problem in XML zu lösen bedient man sich des Konzepts Namensraum (Namesapce). Namensräume stehen in einem eindeutig identifizierbarem Anwendungskontext und dienen dazu Objekte durch Angabe des Objekt-Namens und des Namensraums eindeutig zu identifizieren. So ist es nur erforderlich, dass innerhalb eines Namensraums Objekte eindeutig benannt werden.

### Umsetzung in XML 1.1

Identifiziert werden Namensräume in XML durch Internationalized Resource Identifiers (IRIs), die global eindeutig sind. IRIs sind erweiterte URIs. In XML dürfen nur absolute IRIs verwendet werden. Elemente und Attribute werden durch Präfixe einem Namensraum zugeordnet. Die Deklaration eines Namensraums erfolgt als Attribut eines Elements. Der Namensraum kann über dieses Element hinaus auch in dessen Kind-Elementen verwendet werden.

*Beispiel 1.* Deklaration

```
xmlns:foo="http://www.example.de"
```

## Default-Namensräume

Um nicht jedes mal ein Präfix voranstellen zu müssen gibt es auch die Möglichkeit für ein Element einen Standard-Namensraum fest zu legen.

*Beispiel 2.* Standard-Namensraum

```
xmlns="http://www.foo.org"
```

Elemente ohne Präfix gehören dann zum Standard-Namensraum. Der Standard-Namensraum vererbt sich auch auf die Kind-Elemente. Attribute denen kein Präfix vorangestellt wird, gehören zu dem Namensraum des zugehörigen Elements.

## Gültigkeitsbereich

Namen die einem Namensraum zugeordnet sind, heißen **namensraum-ingeschränkt** oder auch **qualified**. Für eine solche Zuordnung gibt es zwei Möglichkeiten. Entweder legt man einen Standard-Namensraum fest, oder man stellt ein Namensraum-Präfix voran. Element-Namen, die keinem Namensraum zugeordnet sind heißen **namensraumuneingeschränkt** oder auch **unqualified**. Kind-Elemente erben zwar alle Namensraum-Definitionen, doch kann ein geerbter Namensraum lokal neu definiert werden. Eine Neudefinierung mit der leeren IRI löscht die Bindung des Präfixes an einen Namensraum und das Präfix kann bis zur erneuten Bindung nicht verwendet werden.

## Diskussion

Die wesentliche Punkte der Diskussion waren:

- Ohne die Namensräume würde die XML Familie einen großen Teil ihrer Mächtigkeit verlieren.
- Für die Auflösung spielt es keine Rolle auf welche der Möglichkeiten zur Namensraum-Definition benutzt wird, vielmehr ist das Geschmackssache.

---

# Scalable Vector Graphics (SVG)

Tinosch Ganjineh und Christian v. Prollius

## Vortrag

Der Vortrag gliedert sich in die Abschnitte Concepts, Document Structure, Basic Shapes, Filling and Stroking, Gradients, Filter Effects, Animation.

### Concepts

SVG ist eine Auszeichnungssprache zur Beschreibung von zweidimensionalen Vektor Grafiken. Das „S“ steht für scalable und bedeutet, das SVG nicht abhängig von einer festen Bildgröße ist, sondern skalierbare Objekte erzeugt. Das „V“ steht für Vector. Anders als bei pixelorientierten Rasterformaten bestehen geometrische Objekte in SVG aus Linien und Kurven. Das für Graphics stehende „G“ weist auf die erweiterte, sonst nur rudimentäre, Grafikunterstützung in XML hin.

SVG kombiniert verschiedene Grafikobjektarten, wie zum Beispiel Vektorgrafikformen, Bilder und Text. Ein großer Vorteil ist, dass grafische Textelemente von Crawlern erfassbar sind. Im Gegensatz zu Pixelgrafiken, lassen sich bei Vektorgrafiken Objekte ohne Qualitätsverlust zoomen. Darüber hinaus ermöglicht es SVG Animationen zu erstellen und sorgt damit für dynamische Objekte. Als XML Instanz verfügt SVG über ein Document Object Model (DOM) um skriptgesteuerte Interaktionen zu ermöglichen. Weiter ist es möglich mehrere grafische Objekte zu gruppieren und so beispielsweise Transformationen an der ganzen Gruppe vorzunehmen.

Bisher ist Mozilla der einzige Browser, der eine Unterstützung von SVG ermöglicht, doch gibt es von Adobe ein SVG Viewer Plugin.

Der SVG MIME-Typ ist „image/svg+xml“, für den Namensraum hat man „<http://www.w3.org/2000/svg>“.

### Document Structure

Ein SVG Dokument Fragment, ist ein Bereich in einem XML Dokument, der mit `<svg>` beginnt und mit `</svg>` endet. Innerhalb dieses svg Elements

können einzelne Grafikelemente oder Verschachtelungen von Container/Grafik Elementen stehen. Es ist auch möglich dem svg Element keinen Inhalt zu geben. Als eigenständige Ressource oder Datei, bildet ein svg Dokument Fragment ein SVG Dokument.

### Basic Shapes

An einfachen Formen (Basic Shapes) stehen unter SVG Rechtecke, Kreise, Ellipsen, Linien, Polylinien und Polygone zu Verfügung. Bei der Erzeugung solcher Formen, werden neben dem Typ auch Informationen zur Position, Größe, Farbe usw. angegeben.

*Example 1.* Basic Shapes

- `<rect x="400"y="100"width="400"height="200"fill="yellow"stroke="navy"stroke-width="10"/>`
- `<circle cx="600"cy="200"r="100"fill="red"stroke="blue"stroke-width="10"/>`
- `<ellipse transform="translate(900 200) rotate(-30)"rx="250"ry="100"fill="none"stroke="blue"stroke-width="20"/>`
- `<line x1="100"y1="300"x2="300"y2="100"stroke/width="5"/>`
- `polyline fill="none"stroke="blue"stroke-width="10"points="50,375 150,375 150,325 250,325 250,375 350,375"/>`
- `polygon fill="red"stroke="blue"stroke/width="10"points="350,75 379,161 469,161 397,215 423,301 350,250 277,301 303,215 231,161 321,161"/>`

### Filling and Stroking

Mit **fill** lassen sich grafische Elemente oder Text mit Farbe füllen. **fill-rule** gibt an was das „Äußere“ bzw. das „Innere“ einer Form ist.

**stroke** zeichnet ein Rahmen um ein grafisches Element oder um Text. **stroke-width** gibt die Rahmendicke an, mit **stroke-linecap** definiert man die Form des Rahmens.

### Gradients

Die Verwendung von Farbverläufen wird an einem Beispiel demonstriert.

*Example 2.* Gradient

```
...
<linearGradient id="MyGradient">
<stop offset = "5%"stop-color = "F60"/>
<stop offset = "95%"stop-color = "FF6"/>
</linearGradient>
...
```

## Filter Effects

Der Vorteil bei der Verwendung von Filtern liegt in der flexiblen Gestaltung. So lassen sich auf ein Objekt mehrere Filter anwenden wobei die Ursprungsgrafik stets erhalten bleibt. Auch eine Clientseitige Anwendung von Filtern ist möglich.

In dem Vortrag wird an einem Beispiel die Verwendung von Filtern und Animationen erläutert.

## Diskussion

Die wesentliche Punkte der Diskussion waren:

- Farben lassen sich auch als Hexadezimalzahlen angeben.
- SVG ist ein zwar ein Standard, der vom w3c weiter entwickelt wird, trotzdem ist die Unterstützung sehr eingeschränkt. Bisläng ist Mozilla der einzige Browser, der SVG unterstützt. Von Adobe gibt es aber Plugins.
- Es ist nicht vorgesehen den Quelltext zu schützen. Ein Browser muss in der Lage sein den Quelltext zu lesen.
- Da sich die Objekte in SVG auf Container beziehen, ist die Positionierung relativ.



---

# Synchronized Multimedia Integration Language (SMIL)

Georg Sisow und Mattias Hilliges

## Vortrag

### Einordnung in die Weblandschaft

Im Laufe der Zeit verlangte das Web nach mehr Inhalten, so wollte man gerne in der Lage sein, Audio und Video wiederzugeben. Hervorgegangen ist SMIL (Synchronized Multimedia Integration Language) aus den Bemühungen Multimediainhalte in einem Standard zu vereinheitlichen. SMIL ist eine auf XML basierende Markup Sprache und ist für zeitsynchronisierte Multimedia-Präsentationen gedacht. Standardisiert wird SMIL vom W3C (<http://www.w3.org/Audio/Video/>).

### Was bietet SMIL?

SMIL bietet eine reichhaltige Palette an Ausdrucksmöglichkeiten gruppiert in Module für verschiedene Zwecke. Die Präsentationen sind Player ungebunden. Es ist ohne Programmierkenntnisse mögliche einzelne Medien zeitgebunden anzuordnen. SMIL Dateien sind Textdateien und haben die Endung smil. Die abzuspielenden Dateien, sind von verschiedenen Orten verwendbar. Das verwenden von „Container“-Dateien ist überflüssig. Darüber hinaus ermöglicht SMIL Interaktionsmöglichkeiten und unterstützt umgebungsspezifische Entscheidungen.

### SMIL als XML Dokument

Als ein auf XML basierender Standard besitzt SMIL eine XML-DTD. Die Syntaxregeln von XML gelten auch unter SMIL. SMIL ist modular, in logisch zusammengehörigen Bereichen aufgebaut. Es ist möglich einzelne Module zu Profilen zu kombinieren, was eine sinnvolle Auswahl unterstützt. Im Vortrag wurde eine Beispiel Präsentation gezeigt.

## Die technische Seite von SMIL

In SMIL wird durch sequentielle und parallele Ausführung unterschieden. Die sequentielle Abfolge wird mit `<seq>` eine parallele mit `<par>` eingeleitet. Nun sollen einige Attribute genauer betrachtet werden:

### *fit-Attribut*

Regionen in denen Objekte dargestellt werden, werden durch das `<region>` Tag definiert. Das `fit`-Attribut des `<region>` Tag steuert die Anpassung des Objekt an die Region. Das `fit`-Attribut kann verschieden Werte haben.

- **hidden** Das Objekt wird in der Originalgröße angezeigt. Ist das Objekt kleiner als der zu Verfügung stehende Platz, gibt es leere Stellen, andernfalls wird das Objekt beschnitten.
- **fill** Höhe und Breite des Objektes werden unabhängig von einander verändert, so dass das Objekt so groß ist, wie die Region.
- **meet** Das ursprüngliche Größenverhältnis von Höhe und Breite bleibt bestehen. Die Größe des Objekts wird so verändert, dass das Objekt genauso hoch oder genauso breit wie die Region ist und nichts von dem Bild abgeschnitten werden muß.
- **slice** Das ursprüngliche Größenverhältnis von Höhe und Breite bleibt bestehen. Die Größe des Objekts wird so verändert, dass es genauso hoch oder genauso breit wie die Region ist und die Region auf jeden Fall ausgefüllt wird. Der Rest wird abgeschnitten.

### *begin-Attribut*

Das `begin`-Attribut legt den Zeitpunkt fest, an dem ein dynamisches Objekt startet.

### *z-index*

Wurden mehrere Objekte definiert, kann der Wunsch aufkommen die Objekte übereinander zu legen. Mit Hilfe des `z-index`, ein Attribut des `region`-Elements, ist dies möglich. Der `z-index` gibt die Positionierung auf der `z`-Achse an. Regionen mit höherem `z-index` liegen über Regionen mit einem niedrigeren `z-index`.

### *switch-Element*

Mit dem `switch`-Tag kann die Wiedergabe einer SMIL Präsentation gesteuert werden. Für die Elemente zwischen dem `switch`-Tag wird dasjenige ausgewählt, dessen Attribute einen wahren Wert liefern. Wird keine Bedingung wahr, wird auch nichts ausgeführt. Gibt man mehrere Bedingungen an, müssen alle Bedingungen wahr werden, damit die entsprechende Alternative ausgewählt wird. Als eine Auswahl von Attributen für das `switch` Element sind zu nennen:

- **systemBitrate** Übertragungsrate der Verbindung
- **systemScreenSize** Auflösung
- **systemScreenDepth** Farbtiefe
- **systemLanguage** eingestellte Sprache
- **systemOperatingSystem** Betriebssystem

### Anwendungsbereich

Gedacht ist SMIL für interaktive multimediale Präsentationen. Die Präsentationen werden entweder manuell mit Autorenwerkzeugen erstellt oder dynamisch mit Content-Management-Systemen, ähnlich wie bei XML basierten Web-Seiten.

Das SMIL jedoch kaum verbreitet ist mag wohl an der verwirrenden Terminologie, mit Modulen und Profilen liegen aber auch an der momentan verfügbaren Literatur, in der man eine ausführliche Darstellung des Kommunikationspotentials und dem Beitrag von SMIL zu den Medien meist vermisst.

### Diskussion

Die wesentliche Punkte der Diskussion waren:

- SMIL wird von Real-Player unterstützt, allerdings nur teilweise.
- Eine google Suche brachte nur eine SMIL Vorlesung, der Grund sind hohe Kosten und großer Aufwand.
- SMIL kombiniert Multimedia und HTML.
- SMIL ist in erster Linie für den offline Gebrauch gedacht.

