Einführung in die Medieninformatik

Studiengang Medien- und Kommunikationsinformatik

WS 17/18

Prof. Dr.-Ing. Ido A. Iurgel, M.A.

Zum ersten Mal hier?

Willkommen!

gut zu wissen

Sprechzeiten

Meine: Montags 13:15 (ab nächste Woche)

http://www.hochschule-rhein-waal.de/de/panel/sprechzeit**en**

Vertrauensprofessor: Prof. Zimmer

Student Service Center, SSC (Organisatorische Fragen)

Prüfungsordnung (Rahmen-PO, MuKi-PO):

http://www.hochschule-rhein-waal.de/de/hochschule/organisation/satzungen-und-ordnungen/pruefungsordnungen

gut zu wissen

Psychologische Beratung

http://www.hochschule-rhein-waal.de/de/hochschule/service/psychologische-beratung

Prüfungsausschuss

http://www.hochschule-rhein-waal.de/de/fakultaeten/kommunikation-und-umwelt/organisation/pruefungsausschuss

Dekanat

http://www.hochschule-rhein-waal.de/de/fakultaeten/kommunikation-und-umwelt/organisation/dekanat

gut zu wissen

MediaLab II

→ viele Interaktions-Technologien zum Ausprobieren, nach Absprache mit

→ Dr. Thomas Laubach, wissenschaftlicher Mitarbeiter.

→ Freies Arbeiten nach Absprache möglich, Öffnungszeiten werden noch an der Tür hängen.

FabLab

Modalitäten

- Testat
  - bestanden oder nicht bestanden
  - keine Noten
- Übungen und Projektarbeit
  - Beide müssen bestanden werden.

Modalitäten

Der Übungsteil besteht aus Übungsaufgaben.

- Die Aufgaben werden in der Vorlesung bekannt gegeben und werden in Moodle zur Verfügung gestellt.
- Wenn nicht anders angegeben, müssen die Aufgaben bis zum nächsten Montag gelöst und in Moodle hochgeladen werden.
- Eine Aufgabe ist ausreichend gelöst, wenn ein bestimmter Anteil der Punkte / der Aufgabe gelöst sind (meist 60%)
- Sie haben zwei “Freischüsse”: höchstens die Aufgaben von zwei Wochen dürfen nicht abgegeben worden sein oder unter der Bestehensgrenze liegen.
  - Ausnahmen nur in besonderen, belegbaren Fällen möglich
  - Technische Probleme oder ähnliches bei Ihnen begründen

Modalitäten

- Tip: Nutzen Sie die “Freischüsse” nicht ohne zwingenden Grund; diese können Ihnen im Notfall fehlen.
- Die Übungen werden nicht korrigiert zurückgegeben. Es werden die richtigen Lösungen in der Übung besprochen.
- In den Übungsstunden werden Studierende (in der Regel nach dem Zufallsverfahren) gebeten, einzelne Übungen zu erklären. Damit wird überprüft, dass die Übung tatsächlich vom Studierenden erstellt wurde. Sollte das beim Erklären nicht plausibel werden oder der/die Studierende abwesend sein, so wird die gesamte Wochenübung als “nicht bestanden” bewertet.
  - Bei vorheriger Ankündigung kann das Aufrufen auch in der Vorlesung stattfinden.

Modalitäten

- Projektarbeit
  - Im Anschluss an die Übungsaufgaben folgt eine Projektarbeit.
  - Programmieren in Processing (Java) in Gruppen von 4-5 Studierenden
  - Dazu gehören Berichte und zwei Präsentationen.
    - Die Leistung des Einzelnen muss klar erkennbar sein.
    - Details werden noch bekannt gegeben.
  - Die Resultate müssen zum Schluss des Semesters vorgestellt werden. Der Termin wird noch

ABC-Spiel

Was ist Medieninformatik? Assoziieren Sie frei! Für jeden Buchstaben einen Begriff!

A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Z
Y

ABC-Spiel

Digital, Domäne, Daten, Elektronen, E-Government, E-Commerce, E-Inclusion, E-Learning, Fourier-Transformation, Fotografie, Font, Flash, Farbe, GPU, Grafik, GIF, HTML, Head Mounted Display, Hypertext, Hurenkind, Information, Informatik, Interaktivität, Java, JavaScript, JPEG, Klang, Kommunikation, Kanal, Layout, Licht, Link, Maus, Micro..., Medien, Netzwerk, Nachricht, NULL, Operation, OpenGL, Ohr, Prozessor, Pixel, Quantisierung, Radiosity, Rendering, Raster, Realismus, Schall, Schrift, Sampling, Sehen, Story, Transistor, Ton, Tracking, Theorie, Transformation, URL, Usability, Video, Vektor, visuell, Web, Wellen, WWW, XML, Zeichen, Z-Achse, Y-Achse

Literatur

Rainer Malaka, Andreas Butz, Heinrich Hußmann Medieninformatik - Eine Einführung, Pearson Studium 2009

Jennifer Burg, The Science of Digital Media, Prentice Hall 2008

Weitere Literatur und Quellen werden im Verlauf der Vorlesung angegeben.

Was ist *Medieninformatik?*

- Der Begriff *Medieninformatik* ist in der Praxis nicht sehr scharf eingegrenzt.
  - Vgl. die teils unterschiedlichen Inhalte der Textbücher.
- Sie ist ein Zweig der *Informatik.*
  - *Computerwissenschaft, Softwarewissenschaft, Aufbau und Anwendungen des Computers.*
- Digitale Medien hängen mit der *Sinneswahrnehmung* zusammen.
  - Vor allem Sehen, Hören
  - Auch die *Interaktion* mit digitalen Systemen gehört hinein.
  - Digitale Bilder, Video, Animationen, Computerspiele, Musik, Sprache, Geräusche, digitale Texte, Websites, ...

Was ist *Medieninformatik?*

DEFINITIONSVERSUCHE

"Als **digitale Medien** bezeichnet man Informationsträger bzw. Informationstypen, die im Raum und/oder zur Zeit korrelierte Bestandteile haben und innerhalb digitaler Wertebereiche codiert sind."

"**Medieninformatik** ist die Wissenschaft, die sich mit den theoretischen und technischen Grundlagen, der Ver- und Bearbeitung, der Übertragung sowie der Präsentation *digitaler Medien* mit den Mitteln und Methoden der Informatik beschäftigt."

Nachbarn der Medieninformatik

Die Medieninformatik ist stark interdisziplinär.

- Psychologie / Kognitionswissenschaften
- Gestaltung / Design
- Geisteswissenschaften / Medienwissenschaften
- Betriebswirtschaft / Wirtschaftsinformatik
- Usability
- ...

Vektorgrafik

Quelle:
http://clipart-library.com/free-cartoon-graphics.html

Als Vorbereitung: XML

- *eXtensible Markup Language*
- *https://www.w3.org/XML/*
- Sehr weit verbreitete, wichtigste generische “Auszeichnungssprache”.
- Einige Regeln, die wohlgeformte XML-Dokumente auszeichnen:
  - XML fängt meist mit einer Präambel an. Sie gibt zum Beispiel die Version an und ist fakultativ.
    <?xml version="1.0" encoding="UTF-8"?>
  - Es gibt immer ein Wurzelelement, zum Beispiel
    <zoo>
    </zoo>
  - Darin sind weitere Elemente (“Tags”) verschachtelt
    <zoo>

Als Vorbereitung: XML

```
<zoo>
    <tieren>
        <affen>
            <affe alter=”12” name=”Guenther”/>
        </affen>
    </tieren>
    <pfleger>
```

XML

- Elemente können auch Attribute enthalten. Diese gehören immer zum öffnenden Tag:

  <giraffe name="linda" alter="34">

- Elemente haben immer einen öffnenden und einen schließenden Tag.

  <zoo>

  </zoo>

XML

- XML-Dokumente bilden Hierarchien:

```
<zoo>
    <tiere>
        <giraffe/>
    </tiere>
</zoo>
```

- Zwischen öffnendem und schließendem Tag kann Text stehen

```
<beschreibung>
        Das ist ein imaginärer Zoo.
```

XML

BEISPIEL

```
<?xml version="1.0" encoding="UTF-8"?>
<zoo>
    <beschreibung>
        das ist ein imaginärer Zoo
    </beschreibung>
    <tiere>
        <giraffen>
            <giraffe name="linda" alter="34" />
            <giraffe name="paul" alter="4" />
        </giraffen>
        <affen>
            <affe name="krull" alter="6" />
            <affe name="kroll" alter="16" />
```

Einführung in die Medieninformatik

Studiengang Medien- und Kommunikationsinformatik

WS 17/18

Prof. Dr.-Ing. Ido A. Iurgel, M.A.

Zum ersten Mal hier?

Willkommen!

gut zu wissen

Sprechzeiten

Meine: Montags 13:15 (ab nächste Woche)

http://www.hochschule-rhein-waal.de/de/panel/sprechzeit**en**

Vertrauensprofessor: Prof. Zimmer

Student Service Center, SSC (Organisatorische Fragen)

Prüfungsordnung (Rahmen-PO, MuKi-PO):

http://www.hochschule-rhein-waal.de/de/hochschule/organisation/satzungen-und-ordnungen/pruefungsordnungen

gut zu wissen

Psychologische Beratung

http://www.hochschule-rhein-waal.de/de/hochschule/service/psychologische-beratung

Prüfungsausschuss

http://www.hochschule-rhein-waal.de/de/fakultaeten/kommunikation-und-umwelt/organisation/pruefungsausschuss

Dekanat

http://www.hochschule-rhein-waal.de/de/fakultaeten/kommunikation-und-umwelt/organisation/dekanat

gut zu wissen

MediaLab II

→ viele Interaktions-Technologien zum Ausprobieren, nach Absprache mit

→ Dr. Thomas Laubach, wissenschaftlicher Mitarbeiter.

→ Freies Arbeiten nach Absprache möglich, Öffnungszeiten werden noch an der Tür hängen.

FabLab

Modalitäten

- Testat
  - bestanden oder nicht bestanden
  - keine Noten
- Übungen und Projektarbeit
  - Beide müssen bestanden werden.

Modalitäten

Der Übungsteil besteht aus Übungsaufgaben.

- Die Aufgaben werden in der Vorlesung bekannt gegeben und werden in Moodle zur Verfügung gestellt.
- Wenn nicht anders angegeben, müssen die Aufgaben bis zum nächsten Montag gelöst und in Moodle hochgeladen werden.
- Eine Aufgabe ist ausreichend gelöst, wenn ein bestimmter Anteil der Punkte / der Aufgabe gelöst sind (meist 60%)
- Sie haben zwei “Freischüsse”: höchstens die Aufgaben von zwei Wochen dürfen nicht abgegeben worden sein oder unter der Bestehensgrenze liegen.
  - Ausnahmen nur in besonderen, belegbaren Fällen möglich
  - Technische Probleme oder ähnliches bei Ihnen begründen

Modalitäten

- Tip: Nutzen Sie die “Freischüsse” nicht ohne zwingenden Grund; diese können Ihnen im Notfall fehlen.
- Die Übungen werden nicht korrigiert zurückgegeben. Es werden die richtigen Lösungen in der Übung besprochen.
- In den Übungsstunden werden Studierende (in der Regel nach dem Zufallsverfahren) gebeten, einzelne Übungen zu erklären. Damit wird überprüft, dass die Übung tatsächlich vom Studierenden erstellt wurde. Sollte das beim Erklären nicht plausibel werden oder der/die Studierende abwesend sein, so wird die gesamte Wochenübung als “nicht bestanden” bewertet.
  - Bei vorheriger Ankündigung kann das Aufrufen auch in der Vorlesung stattfinden.

Modalitäten

- Projektarbeit
  - Im Anschluss an die Übungsaufgaben folgt eine Projektarbeit.
  - Programmieren in Processing (Java) in Gruppen von 4-5 Studierenden
  - Dazu gehören Berichte und zwei Präsentationen.
    - Die Leistung des Einzelnen muss klar erkennbar sein.
    - Details werden noch bekannt gegeben.
  - Die Resultate müssen zum Schluss des Semesters vorgestellt werden. Der Termin wird noch

ABC-Spiel

Was ist Medieninformatik? Assoziieren Sie frei! Für jeden Buchstaben einen Begriff!

A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Z
Y

ABC-Spiel

Digital, Domäne, Daten, Elektronen, E-Government, E-Commerce, E-Inclusion, E-Learning, Fourier-Transformation, Fotografie, Font, Flash, Farbe, GPU, Grafik, GIF, HTML, Head Mounted Display, Hypertext, Hurenkind, Information, Informatik, Interaktivität, Java, JavaScript, JPEG, Klang, Kommunikation, Kanal, Layout, Licht, Link, Maus, Micro..., Medien, Netzwerk, Nachricht, NULL, Operation, OpenGL, Ohr, Prozessor, Pixel, Quantisierung, Radiosity, Rendering, Raster, Realismus, Schall, Schrift, Sampling, Sehen, Story, Transistor, Ton, Tracking, Theorie, Transformation, URL, Usability, Video, Vektor, visuell, Web, Wellen, WWW, XML, Zeichen, Z-Achse, Y-Achse

Literatur

Rainer Malaka, Andreas Butz, Heinrich Hußmann Medieninformatik - Eine Einführung, Pearson Studium 2009

Jennifer Burg, The Science of Digital Media, Prentice Hall 2008

Weitere Literatur und Quellen werden im Verlauf der Vorlesung angegeben.

Was ist *Medieninformatik?*

- Der Begriff *Medieninformatik* ist in der Praxis nicht sehr scharf eingegrenzt.
  - Vgl. die teils unterschiedlichen Inhalte der Textbücher.
- Sie ist ein Zweig der *Informatik.*
  - *Computerwissenschaft, Softwarewissenschaft, Aufbau und Anwendungen des Computers.*
- Digitale Medien hängen mit der *Sinneswahrnehmung* zusammen.
  - Vor allem Sehen, Hören
  - Auch die *Interaktion* mit digitalen Systemen gehört hinein.
  - Digitale Bilder, Video, Animationen, Computerspiele, Musik, Sprache, Geräusche, digitale Texte, Websites, ...

Was ist *Medieninformatik?*

DEFINITIONSVERSUCHE

"Als **digitale Medien** bezeichnet man Informationsträger bzw. Informationstypen, die im Raum und/oder zur Zeit korrelierte Bestandteile haben und innerhalb digitaler Wertebereiche codiert sind."

"**Medieninformatik** ist die Wissenschaft, die sich mit den theoretischen und technischen Grundlagen, der Ver- und Bearbeitung, der Übertragung sowie der Präsentation *digitaler Medien* mit den Mitteln und Methoden der Informatik beschäftigt."

Nachbarn der Medieninformatik

Die Medieninformatik ist stark interdisziplinär.

- Psychologie / Kognitionswissenschaften
- Gestaltung / Design
- Geisteswissenschaften / Medienwissenschaften
- Betriebswirtschaft / Wirtschaftsinformatik
- Usability
- ...

Vektorgrafik

Quelle:
http://clipart-library.com/free-cartoon-graphics.html

Als Vorbereitung: XML

- *eXtensible Markup Language*
- *https://www.w3.org/XML/*
- Sehr weit verbreitete, wichtigste generische “Auszeichnungssprache”.
- Einige Regeln, die wohlgeformte XML-Dokumente auszeichnen:
  - XML fängt meist mit einer Präambel an. Sie gibt zum Beispiel die Version an und ist fakultativ.
    <?xml version="1.0" encoding="UTF-8"?>
  - Es gibt immer ein Wurzelelement, zum Beispiel
    <zoo>
    </zoo>
  - Darin sind weitere Elemente (“Tags”) verschachtelt
    <zoo>

Als Vorbereitung: XML

```
<zoo>
    <tieren>
        <affen>
            <affe alter=”12” name=”Guenther”/>
        </affen>
    </tieren>
    <pfleger>
```

XML

- Elemente können auch Attribute enthalten. Diese gehören immer zum öffnenden Tag:

  <giraffe name="linda" alter="34">

- Elemente haben immer einen öffnenden und einen schließenden Tag.

  <zoo>

  </zoo>

XML

- XML-Dokumente bilden Hierarchien:

```
<zoo>
    <tiere>
        <giraffe/>
    </tiere>
</zoo>
```

- Zwischen öffnendem und schließendem Tag kann Text stehen

```
<beschreibung>
        Das ist ein imaginärer Zoo.
```

XML

BEISPIEL

```
<?xml version="1.0" encoding="UTF-8"?>
<zoo>
    <beschreibung>
        das ist ein imaginärer Zoo
    </beschreibung>
    <tiere>
        <giraffen>
            <giraffe name="linda" alter="34" />
            <giraffe name="paul" alter="4" />
        </giraffen>
        <affen>
            <affe name="krull" alter="6" />
            <affe name="kroll" alter="16" />
```

Quiz

```
<tiere>
    <katzen>
        <katze name=”lulu” gewicht=”4kg”/>
        <katze name=”frodo” gewicht=”6kg” />
    </katzen>
</tiere>
```

Auf Deutsch: Die einzelne Katze Lulu hat ein Gewicht von 4kg, und die Katze Frodo hat ein Gewicht von 6kg, und Lulu und Frodo

Quiz

```
<autos>
    <weiss>
        <auto kennzeichen=”E4354” besitzer=”lisa müller ” />
    </weiss>
    <rot>
        <auto kennzeichen=”KL2344” besitzer=”paul schmidt ” />
    </rot>
</autos>
```

Quiz

Finden Sie syntaktische Fehler - achten Sie nicht auf den Inhalt oder darauf, ob das Dokument Sinn ergibt!

```
<zoo>
  <tiere>
        <giraffen>
            <giraffe name=”linda” alter=”34”/>
            <Giraffe name=”paul” alter=”42”/>
         </giraffen>
    <affen>
            <affe name=”krull” alter=”6”/>
```

Vektorgrafik: SVG

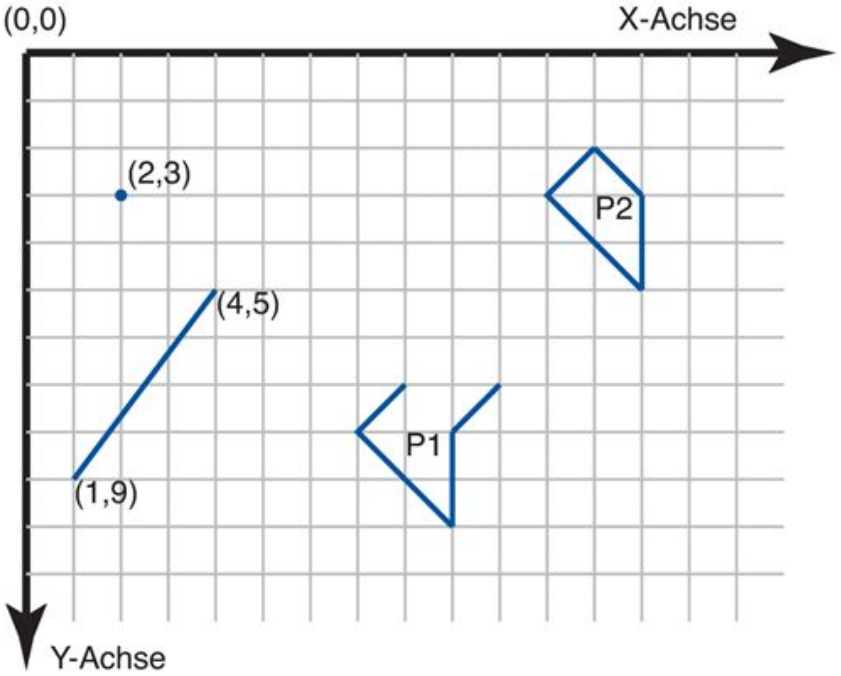
Referenzen zu SVG:

http://www.w3.org/Graphics/SVG/IG/resources/svgprimer.html,

Vektorgrafik

- Vektorgrafik: Anstelle von Pixeln verwendet man eine Beschreibungssprache für grundlegende Elementen wie z.B. Kurven, Kreise, Geraden, und Polygone.
- Vektorgrafiken lassen sich beliebig groß oder klein darstellen.
- Vektorgrafiken brauchen immer gleich viel Speicherplatz, unabhängig von der Größe der Darstellung
- Nicht alle Grafiken lassen sich sinnvoll als Vektorgrafiken darstellen. Am besten eignen sich Grafiken, die von Anfang an als Vektorgrafiken konzipiert wurden. Schriften und viele Plakate sind gute Beispiele. Photos lassen sich selten sinnvoll

Vektorgrafik



Punkt,
Gerade,
Polygon

Abbildung aus: *Rainer Malaka, Andreas Butz, Heinrich Hußmann: Medieninformatik - Eine*

Vektorgrafik: SVG

- SVG: Scalable Vector Graphics. Web-Standard des W3C für Darstellung von Vektorgrafiken
  - W3C ist das World Wide Web Consortium. Es entwickelt Standards für das Internet.
- Wird von allen gängigen Browsern unterstützt (nicht unbedingt in jeder Hinsicht).
- SVG ist eine XML-Sprache.

Vektorgrafik: SVG

REINES SVG:

```
<svg xmlns="http://www.w3.org/2000/svg" width="200" height="100"
version="1.1">
  <line x1="0" y1="100" x2="100" y2="0"  stroke-width="2" stroke="black" />
</svg>
```

EINGEBETTET IN HTML:

```
<!DOCTYPE html>
<html>
      <body>
        <svg xmlns="http://www.w3.org/2000/svg" width="200" height="100"
                version="1.1">
          <line x1="0" y1="10" x2="100" y2="0" stroke-width="2" stroke="black" />
```

SVG: Polyline

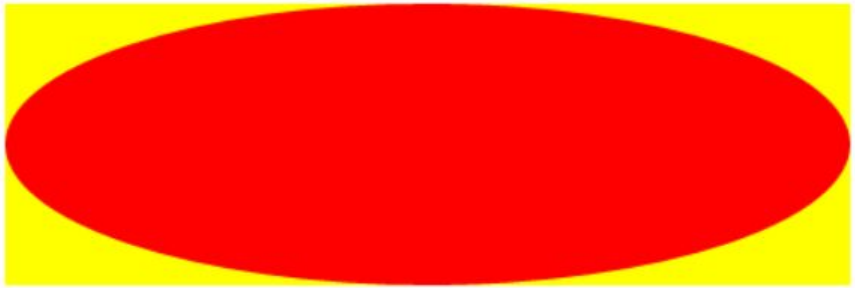
```
<svg width="120" height="120" xmlns="http://www.w3.org/2000/svg">
  <polyline fill="none" stroke="black"
      points="20,100 40,60 70,80 100,20"/>
</svg>
```

SVG: Polygon

```
<svg width="120" height="120" xmlns="http://www.w3.org/2000/svg">
  <polygon fill="none" stroke="black"
     points="20,100 40,60 70,80 100,20"/>
</svg>
```

SVG: Polygon

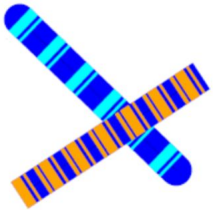
```
<svg width="320" height="420" xmlns="http://www.w3.org/2000/svg">
    <rect x="0" y="0" width="200" height="100" fill="#00FF00" />
    <ellipse cx="100" cy="50" rx="100" ry="50" fill="red" />
</svg>
```



Nach: https://www.w3.org/Graphics/SVG/IG/resources/svgprimer.html

SVG: Polygon

```
<svg width="320" height="420" xmlns="http://www.w3.org/2000/svg">
    <line x1="15" y1="15" x2="140" y2="135" stroke-width="25" stroke="blue"
        stroke-linecap="round" />

    <line x1="15" y1="15" x2="140" y2="135" stroke-width="25" stroke="aqua"
        stroke-dasharray="8,3,2,18" />

    <line x1="15" y1="155" x2="160" y2="60" stroke-width="25" stroke="blue" />

    <line x1="15" y1="155" x2="160" y2="60" stroke-width="25" stro
        stroke-dasharray="8,3,2" />
    d" />
```



Nach: https://www.w3.org/Graphics/SVG/IG/resources/svgprimer.html

SVG: Path

```
<svg width="320" height="420" xmlns="http://www.w3.org/2000/svg">
    <path d="M 10 20 L 30 70, 75 100" stroke="black" fill="none" stroke-width="5">
</svg>
```

WEITERE BEISPIELE:

```
<path d="M 100 50 L 200 150 100 100" fill="none" stroke="black"/>
<path d="M 100,50 200,150 100,100 z" fill="none" stroke="black"/>
<path d="M 70,290 L 150,150 200,250 40,250 100,150 170,290"
```

Nach: https://www.w3.org/Graphics/SVG/IG/resources/svgprimer.html

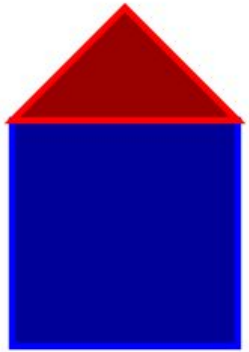
SVG: Circles

```
<svg width="320" height="420" xmlns="http://www.w3.org/2000/svg">
    <circle cx="80" cy="50" r="40" />
    <circle cx="80" cy="110" r="40" fill="red" fill-opacity="0.5" />
    <circle cx="80" cy="170" r="40" fill="yellow" stroke="blue" />
    <circle cx="80" cy="160" r="20" fill="red" stroke="black"
        stroke-width="10" />
    <circle cx="140" cy="110" r="60" fill="none" stroke="#57
        stroke-width="30" stroke-dasharray="3,5,8,13">
</svg>
```

Nach: https://www.w3.org/Graphics/SVG/IG/resources/svgprimer.html

Vektorgrafik: SVG

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
  "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg xmlns="http://www.w3.org/2000/svg">
        <rect stroke="blue" stroke-width="3px" fill="#000099"
x="100" y="100"
                width="100" height="100" />
        <polygon stroke="red" stroke-width="3px" fill="#990000"
points="100,100 200,100 200, 50 100,50" />
</svg>
```



Vektorgrafik: SVG

Beispiele für grundlegende Formen in SVG:

- <rect>
- <circle>
- <ellipse>
- <line>
- <polyline>
- <polygon>
- <path>

Weitere Beispiele für wichtige Elemente:

- Linienart (“stroke”)
- <text>

Vektorgrafik: SVG

Beispiele:

```
1)
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <rect width="300" height="100"
style="fill:rgb(0,0,255);stroke-width:1;stroke:rgb(0,0,0)"/>
</svg>

2)
<svg xmlns="http://www.w3.org/2000/svg" versi
  <rect x="50" y="20" rx="20" ry="20" width="
height="150"
  style="fill:red;stroke:black;stroke-width:5
</svg>
```

.5"/>

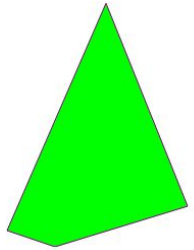
Beispiele aus http://www.w3schools.com/

Vektorgrafik: SVG

Beispiele:

A)

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <line x1="0" y1="0" x2="200" y2="200"
  style="stroke:rgb(255,0,0);stroke-width:2"/>
</svg>
```



B)

```
<svg xmlns="http://www.w3.org/2000/svg"
                                    version="1.1">
  <polygon points="220,10 300,210 170,250 123,234"
  style="fill:lime;stroke:purple;stroke-width:1"/>
</svg>
```

C)

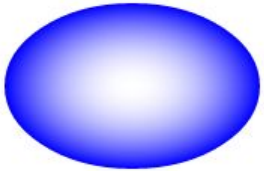
```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
```

Beispiele aus http://www.w3schools.com/

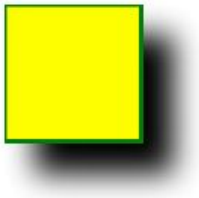
Vektorgrafik: SVG

Weitere Möglichkeiten von SVG (Beispiele):

Gradienten



Schatten



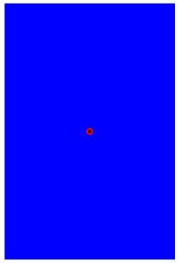
Weichzeichnen

Beispiele aus http://www.w3schools.com/

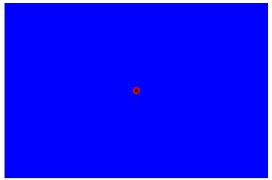
Vektorgrafik: SVG

Transformationen (Translation, Rotation):

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <rect x="50" y="100" width="200" height="300" fill="blue"
  transform="translate(50,  50)"/>
  <circle cx="200" cy="300" r="3" stroke-width="2"
   stroke="red"/>
</svg>
```



```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <rect x="100" y="200" width="200" height="300" fill="blue"
     transform="rotate(100,200,350)"/>
   <circle cx="200" cy="350" r="3" stroke-width="2"
```



Einführung in die Medieninformatik

Studiengang Medien- und Kommunikationsinformatik

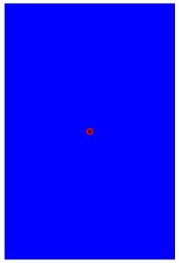
WS 17/18

Prof. Dr.-Ing. Ido A. Iurgel, M.A.

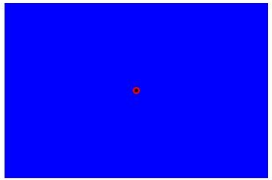
Vektorgrafik: SVG

Transformationen (Translation, Rotation):

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <rect x="50" y="100" width="200" height="300" fill="blue"
  transform="translate(50,  50)"/>
  <circle cx="200" cy="300" r="3" stroke-width="2"
   stroke="red"/>
</svg>
```



```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <rect x="100" y="200" width="200" height="300" fill="blue"
     transform="rotate(100,200,350)"/>
   <circle cx="200" cy="350" r="3" stroke-width="2"
```

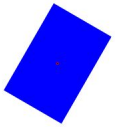


Vektorgrafik: SVG

Transformationen (Skalierung, Verkettung):

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <rect x="100" y="200" width="200" height="300" fill="blue"
     transform="scale(0.3)"/>
   <circle cx="200" cy="350" r="3" stroke-width="2" stroke="red"/>
</svg>
```

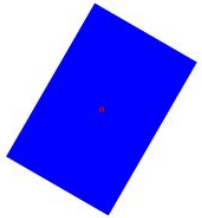
```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <rect x="100" y="200" width="200" height="300" fill="blue"
transform="translate(50,50), rotate(30,200,350),scale(0.5)"/>
   <circle cx="200" cy="350" r="3" stroke-width="2" stroke="red"
```



Vektorgrafik: SVG

Gruppierung <g> - Befehle werden auf alle Elemente der Gruppe angewendet:

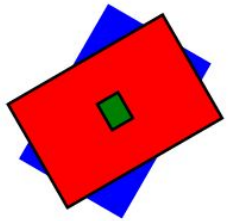
```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
    <g transform="translate(50,50), rotate(30,200,350),scale(0.5)">
        <rect x="100" y="200" width="200" height="300" fill="blue" />
        <circle cx="200" cy="350" r="3" stroke-width="2
    </g>
</svg>
```



Vektorgrafik: SVG

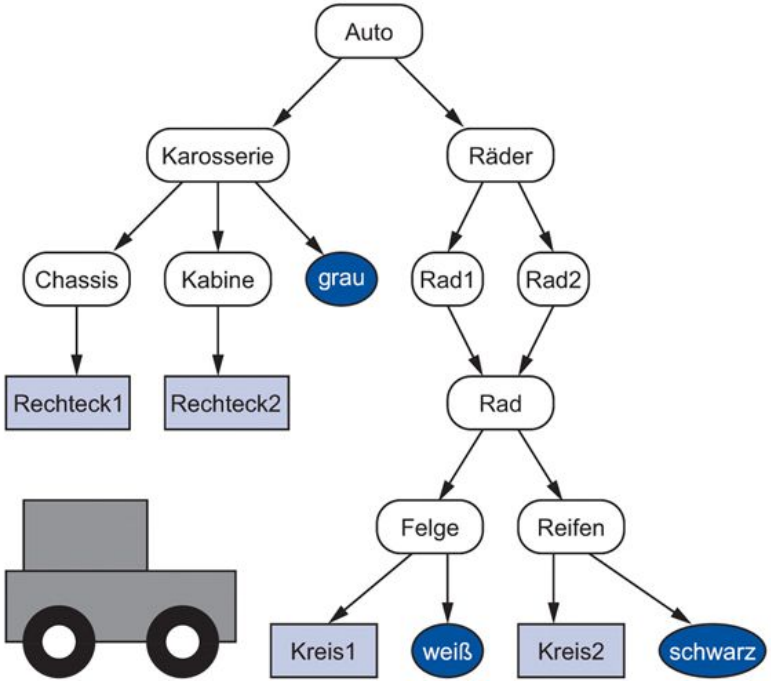
Gruppierung <g> - Gruppen können weitere Gruppen beinhalten:

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
<g transform="translate(50,50), rotate(30,200,350),scale(0.5)">
  <rect x="100" y="200" width="200" height="300" fill="blue" />
   <circle cx="200" cy="350" r="3" stroke-width="2" stroke="red"/>
   <g transform="rotate(30,200,350)" stroke="black" stroke-width  "5"
     <rect x="100" y="200" width="200" height="300" fill="red" />
     <rect x="100" y="200" width="50" height="40" fill="green"
       transform="translate(75,130)"/>
   </g>
</g>
</svg>
```



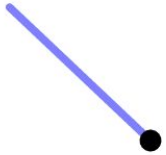
Szenengraphen: 2D und 3D Grafik

Szenengraph
für ein Auto



Vektorgrafik: SVG

SVG unterstützt auch Animationen



animateTransform-Element:

```
<svg xmlns="http://www.w3.org/2000/svg" width="300" height="500">
    <g transform="translate(70 50)" id="hour">
        <line stroke-width="5" x1="0" y1="0" x2="44" y2="44"
            stroke-linecap="round" stroke="blue" opacity=".5" />
        <animateTransform attributeName="transform" type="rotate"
            repeatCount="indefinite" dur="60s" by="360" />
        <circle r="7" />
    </g>
</svg>
```

Einführung in die Medieninformatik

Studiengang Medien- und Kommunikationsinformatik

WS 17/18

Prof. Dr.-Ing. Ido A. Iurgel, M.A.

Zum ersten Mal hier?

Willkommen!

gut zu wissen

Sprechzeiten

Meine: Montags 13:15 (ab nächste Woche)

http://www.hochschule-rhein-waal.de/de/panel/sprechzeit**en**

Vertrauensprofessor: Prof. Zimmer

Student Service Center, SSC (Organisatorische Fragen)

Prüfungsordnung (Rahmen-PO, MuKi-PO):

http://www.hochschule-rhein-waal.de/de/hochschule/organisation/satzungen-und-ordnungen/pruefungsordnungen

gut zu wissen

Psychologische Beratung

http://www.hochschule-rhein-waal.de/de/hochschule/service/psychologische-beratung

Prüfungsausschuss

http://www.hochschule-rhein-waal.de/de/fakultaeten/kommunikation-und-umwelt/organisation/pruefungsausschuss

Dekanat

http://www.hochschule-rhein-waal.de/de/fakultaeten/kommunikation-und-umwelt/organisation/dekanat

gut zu wissen

MediaLab II

→ viele Interaktions-Technologien zum Ausprobieren, nach Absprache mit

→ Dr. Thomas Laubach, wissenschaftlicher Mitarbeiter.

→ Freies Arbeiten nach Absprache möglich, Öffnungszeiten werden noch an der Tür hängen.

FabLab

Modalitäten

- Testat
  - bestanden oder nicht bestanden
  - keine Noten
- Übungen und Projektarbeit
  - Beide müssen bestanden werden.

Modalitäten

Der Übungsteil besteht aus Übungsaufgaben.

- Die Aufgaben werden in der Vorlesung bekannt gegeben und werden in Moodle zur Verfügung gestellt.
- Wenn nicht anders angegeben, müssen die Aufgaben bis zum nächsten Montag gelöst und in Moodle hochgeladen werden.
- Eine Aufgabe ist ausreichend gelöst, wenn ein bestimmter Anteil der Punkte / der Aufgabe gelöst sind (meist 60%)
- Sie haben zwei “Freischüsse”: höchstens die Aufgaben von zwei Wochen dürfen nicht abgegeben worden sein oder unter der Bestehensgrenze liegen.
  - Ausnahmen nur in besonderen, belegbaren Fällen möglich
  - Technische Probleme oder ähnliches bei Ihnen begründen

Modalitäten

- Tip: Nutzen Sie die “Freischüsse” nicht ohne zwingenden Grund; diese können Ihnen im Notfall fehlen.
- Die Übungen werden nicht korrigiert zurückgegeben. Es werden die richtigen Lösungen in der Übung besprochen.
- In den Übungsstunden werden Studierende (in der Regel nach dem Zufallsverfahren) gebeten, einzelne Übungen zu erklären. Damit wird überprüft, dass die Übung tatsächlich vom Studierenden erstellt wurde. Sollte das beim Erklären nicht plausibel werden oder der/die Studierende abwesend sein, so wird die gesamte Wochenübung als “nicht bestanden” bewertet.
  - Bei vorheriger Ankündigung kann das Aufrufen auch in der Vorlesung stattfinden.

Modalitäten

- Projektarbeit
  - Im Anschluss an die Übungsaufgaben folgt eine Projektarbeit.
  - Programmieren in Processing (Java) in Gruppen von 4-5 Studierenden
  - Dazu gehören Berichte und zwei Präsentationen.
    - Die Leistung des Einzelnen muss klar erkennbar sein.
    - Details werden noch bekannt gegeben.
  - Die Resultate müssen zum Schluss des Semesters vorgestellt werden. Der Termin wird noch

ABC-Spiel

Was ist Medieninformatik? Assoziieren Sie frei! Für jeden Buchstaben einen Begriff!

A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Z
Y

ABC-Spiel

Digital, Domäne, Daten, Elektronen, E-Government, E-Commerce, E-Inclusion, E-Learning, Fourier-Transformation, Fotografie, Font, Flash, Farbe, GPU, Grafik, GIF, HTML, Head Mounted Display, Hypertext, Hurenkind, Information, Informatik, Interaktivität, Java, JavaScript, JPEG, Klang, Kommunikation, Kanal, Layout, Licht, Link, Maus, Micro..., Medien, Netzwerk, Nachricht, NULL, Operation, OpenGL, Ohr, Prozessor, Pixel, Quantisierung, Radiosity, Rendering, Raster, Realismus, Schall, Schrift, Sampling, Sehen, Story, Transistor, Ton, Tracking, Theorie, Transformation, URL, Usability, Video, Vektor, visuell, Web, Wellen, WWW, XML, Zeichen, Z-Achse, Y-Achse

Literatur

Rainer Malaka, Andreas Butz, Heinrich Hußmann Medieninformatik - Eine Einführung, Pearson Studium 2009

Jennifer Burg, The Science of Digital Media, Prentice Hall 2008

Weitere Literatur und Quellen werden im Verlauf der Vorlesung angegeben.

Was ist *Medieninformatik?*

- Der Begriff *Medieninformatik* ist in der Praxis nicht sehr scharf eingegrenzt.
  - Vgl. die teils unterschiedlichen Inhalte der Textbücher.
- Sie ist ein Zweig der *Informatik.*
  - *Computerwissenschaft, Softwarewissenschaft, Aufbau und Anwendungen des Computers.*
- Digitale Medien hängen mit der *Sinneswahrnehmung* zusammen.
  - Vor allem Sehen, Hören
  - Auch die *Interaktion* mit digitalen Systemen gehört hinein.
  - Digitale Bilder, Video, Animationen, Computerspiele, Musik, Sprache, Geräusche, digitale Texte, Websites, ...

Was ist *Medieninformatik?*

DEFINITIONSVERSUCHE

"Als **digitale Medien** bezeichnet man Informationsträger bzw. Informationstypen, die im Raum und/oder zur Zeit korrelierte Bestandteile haben und innerhalb digitaler Wertebereiche codiert sind."

"**Medieninformatik** ist die Wissenschaft, die sich mit den theoretischen und technischen Grundlagen, der Ver- und Bearbeitung, der Übertragung sowie der Präsentation *digitaler Medien* mit den Mitteln und Methoden der Informatik beschäftigt."

Nachbarn der Medieninformatik

Die Medieninformatik ist stark interdisziplinär.

- Psychologie / Kognitionswissenschaften
- Gestaltung / Design
- Geisteswissenschaften / Medienwissenschaften
- Betriebswirtschaft / Wirtschaftsinformatik
- Usability
- ...

Als Vorbereitung: XML

- *eXtensible Markup Language*
- *https://www.w3.org/XML/*
- Sehr weit verbreitete, wichtigste generische “Auszeichnungssprache”.
- Einige Regeln, die wohlgeformte XML-Dokumente auszeichnen:
  - XML fängt meist mit einer Präambel an. Sie gibt zum Beispiel die Version an und ist fakultativ.
    <?xml version="1.0" encoding="UTF-8"?>
  - Es gibt immer ein Wurzelelement, zum Beispiel
    <zoo>
    </zoo>
  - Darin sind weitere Elemente (“Tags”) verschachtelt
    <zoo>

Als Vorbereitung: XML

```
<zoo>
    <tieren>
        <affen>
            <affe alter=”12” name=”Guenther”/>
        </affen>
    </tieren>
    <pfleger>
```

XML

- Elemente können auch Attribute enthalten. Diese gehören immer zum öffnenden Tag:

  <giraffe name="linda" alter="34">

- Elemente haben immer einen öffnenden und einen schließenden Tag.

  <zoo>

  </zoo>

XML

- XML-Dokumente bilden Hierarchien:

```
<zoo>
    <tiere>
        <giraffe/>
    </tiere>
</zoo>
```

- Zwischen öffnendem und schließendem Tag kann Text stehen

```
<beschreibung>
        Das ist ein imaginärer Zoo.
```

XML

BEISPIEL

```
<?xml version="1.0" encoding="UTF-8"?>
<zoo>
    <beschreibung>
        das ist ein imaginärer Zoo
    </beschreibung>
    <tiere>
        <giraffen>
            <giraffe name="linda" alter="34" />
            <giraffe name="paul" alter="4" />
        </giraffen>
        <affen>
            <affe name="krull" alter="6" />
            <affe name="kroll" alter="16" />
```

Quiz

```
<tiere>
    <katzen>
        <katze name=”lulu” gewicht=”4kg”/>
        <katze name=”frodo” gewicht=”6kg” />
    </katzen>
</tiere>
```

Auf Deutsch: Die einzelne Katze Lulu hat ein Gewicht von 4kg, und die Katze Frodo hat ein Gewicht von 6kg, und Lulu und Frodo

Quiz

```
<autos>
    <weiss>
        <auto kennzeichen=”E4354” besitzer=”lisa müller ” />
    </weiss>
    <rot>
        <auto kennzeichen=”KL2344” besitzer=”paul schmidt ” />
    </rot>
</autos>
```

Quiz

Finden Sie syntaktische Fehler - achten Sie nicht auf den Inhalt oder darauf, ob das Dokument Sinn ergibt!

```
<zoo>
  <tiere>
        <giraffen>
            <giraffe name=”linda” alter=”34”/>
            <Giraffe name=”paul” alter=”42”/>
         </giraffen>
    <affen>
            <affe name=”krull” alter=”6”/>
```

Vektorgrafik: SVG

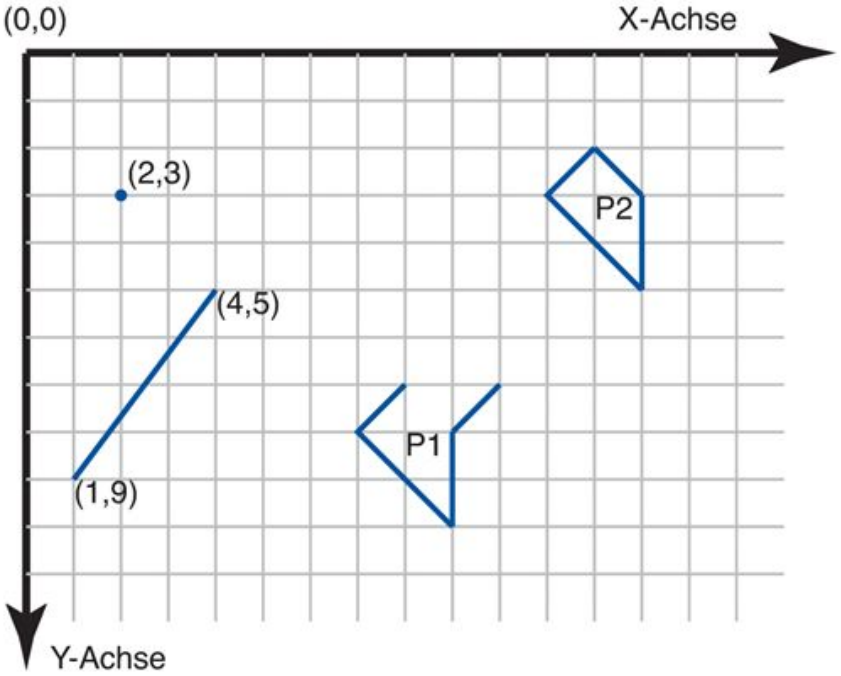
Referenzen zu SVG:

http://www.w3.org/Graphics/SVG/IG/resources/svgprimer.html,

Vektorgrafik

- Vektorgrafik: Anstelle von Pixeln verwendet man eine Beschreibungssprache für grundlegende Elementen wie z.B. Kurven, Kreise, Geraden, und Polygone.
- Vektorgrafiken lassen sich beliebig groß oder klein darstellen.
- Vektorgrafiken brauchen immer gleich viel Speicherplatz, unabhängig von der Größe der Darstellung
- Nicht alle Grafiken lassen sich sinnvoll als Vektorgrafiken darstellen. Am besten eignen sich Grafiken, die von Anfang an als Vektorgrafiken konzipiert wurden. Schriften und viele Plakate sind gute Beispiele. Photos lassen sich selten sinnvoll

Vektorgrafik



Punkt,
Gerade,
Polygon

Abbildung aus: *Rainer Malaka, Andreas Butz, Heinrich Hußmann: Medieninformatik - Eine*

Vektorgrafik: SVG

- SVG: Scalable Vector Graphics. Web-Standard des W3C für Darstellung von Vektorgrafiken
  - W3C ist das World Wide Web Consortium. Es entwickelt Standards für das Internet.
- Wird von allen gängigen Browsern unterstützt (nicht unbedingt in jeder Hinsicht).
- SVG ist eine XML-Sprache.

Vektorgrafik: SVG

REINES SVG:

```
<svg xmlns="http://www.w3.org/2000/svg" width="200" height="100"
version="1.1">
  <line x1="0" y1="100" x2="100" y2="0"  stroke-width="2" stroke="black" />
</svg>
```

EINGEBETTET IN HTML:

```
<!DOCTYPE html>
<html>
      <body>
        <svg xmlns="http://www.w3.org/2000/svg" width="200" height="100"
                version="1.1">
          <line x1="0" y1="10" x2="100" y2="0" stroke-width="2" stroke="black" />
```

SVG: Polyline

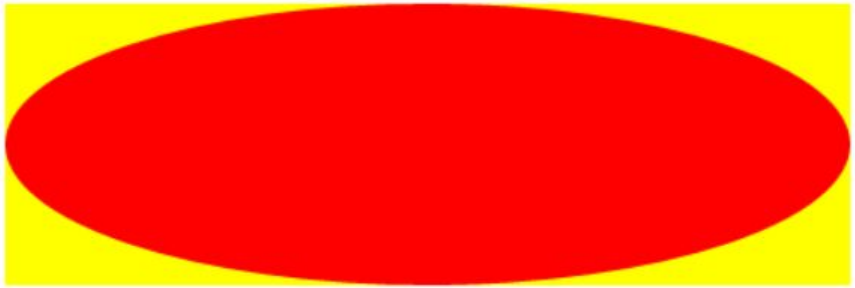
```
<svg width="120" height="120" xmlns="http://www.w3.org/2000/svg">
  <polyline fill="none" stroke="black"
      points="20,100 40,60 70,80 100,20"/>
</svg>
```

SVG: Polygon

```
<svg width="120" height="120" xmlns="http://www.w3.org/2000/svg">
  <polygon fill="none" stroke="black"
     points="20,100 40,60 70,80 100,20"/>
</svg>
```

SVG: Polygon

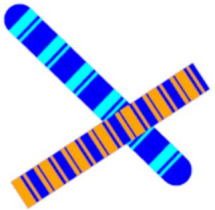
```
<svg width="320" height="420" xmlns="http://www.w3.org/2000/svg">
    <rect x="0" y="0" width="200" height="100" fill="#00FF00" />
    <ellipse cx="100" cy="50" rx="100" ry="50" fill="red" />
</svg>
```



Nach: https://www.w3.org/Graphics/SVG/IG/resources/svgprimer.html

SVG: Polygon

```
<svg width="320" height="420" xmlns="http://www.w3.org/2000/svg">
    <line x1="15" y1="15" x2="140" y2="135" stroke-width="25" stroke="blue"
        stroke-linecap="round" />

    <line x1="15" y1="15" x2="140" y2="135" stroke-width="25" stroke="aqua"
        stroke-dasharray="8,3,2,18" />

    <line x1="15" y1="155" x2="160" y2="60" stroke-width="25" stroke="blue" />

    <line x1="15" y1="155" x2="160" y2="60" stroke-width="25" stro
        stroke-dasharray="8,3,2" />
    d" />
```



Nach: https://www.w3.org/Graphics/SVG/IG/resources/svgprimer.html

SVG: Path

```
<svg width="320" height="420" xmlns="http://www.w3.org/2000/svg">
    <path d="M 10 20 L 30 70, 75 100" stroke="black" fill="none" stroke-width="5">
</svg>
```

WEITERE BEISPIELE:

```
<path d="M 100 50 L 200 150 100 100" fill="none" stroke="black"/>
<path d="M 100,50 200,150 100,100 z" fill="none" stroke="black"/>
<path d="M 70,290 L 150,150 200,250 40,250 100,150 170,290"
```

Nach: https://www.w3.org/Graphics/SVG/IG/resources/svgprimer.html

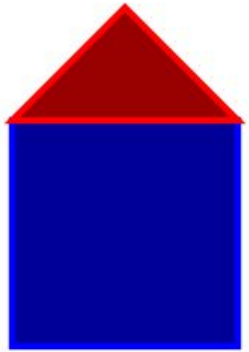
SVG: Circles

```
<svg width="320" height="420" xmlns="http://www.w3.org/2000/svg">
    <circle cx="80" cy="50" r="40" />
    <circle cx="80" cy="110" r="40" fill="red" fill-opacity="0.5" />
    <circle cx="80" cy="170" r="40" fill="yellow" stroke="blue" />
    <circle cx="80" cy="160" r="20" fill="red" stroke="black"
        stroke-width="10" />
    <circle cx="140" cy="110" r="60" fill="none" stroke="#57
        stroke-width="30" stroke-dasharray="3,5,8,13">
</svg>
```

Nach: https://www.w3.org/Graphics/SVG/IG/resources/svgprimer.html

Vektorgrafik: SVG

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
  "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg xmlns="http://www.w3.org/2000/svg">
        <rect stroke="blue" stroke-width="3px" fill="#000099"
x="100" y="100"
                width="100" height="100" />
        <polygon stroke="red" stroke-width="3px" fill="#990000"
points="100,100 200,100 200, 50 100,50" />
</svg>
```



Vektorgrafik: SVG

Beispiele für grundlegende Formen in SVG:

- <rect>
- <circle>
- <ellipse>
- <line>
- <polyline>
- <polygon>
- <path>

Weitere Beispiele für wichtige Elemente:

- Linienart (“stroke”)
- <text>

Vektorgrafik: SVG

Beispiele:

```
1)
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <rect width="300" height="100"
style="fill:rgb(0,0,255);stroke-width:1;stroke:rgb(0,0,0)"/>
</svg>

2)
<svg xmlns="http://www.w3.org/2000/svg" versi
  <rect x="50" y="20" rx="20" ry="20" width="
height="150"
  style="fill:red;stroke:black;stroke-width:5
</svg>
```

.5"/>

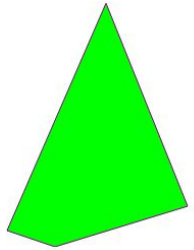
Beispiele aus http://www.w3schools.com/

Vektorgrafik: SVG

Beispiele:

A)

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <line x1="0" y1="0" x2="200" y2="200"
  style="stroke:rgb(255,0,0);stroke-width:2"/>
</svg>
```



B)

```
<svg xmlns="http://www.w3.org/2000/svg"
                                    version="1.1">
  <polygon points="220,10 300,210 170,250 123,234"
  style="fill:lime;stroke:purple;stroke-width:1"/>
</svg>
```

C)

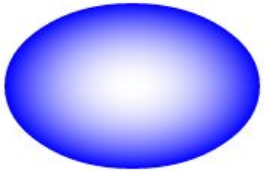
```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
```

Beispiele aus http://www.w3schools.com/

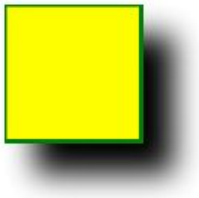
Vektorgrafik: SVG

Weitere Möglichkeiten von SVG (Beispiele):

Gradienten



Schatten



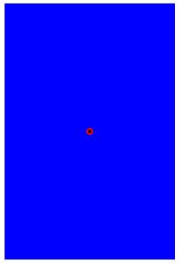
Weichzeichnen

Beispiele aus http://www.w3schools.com/

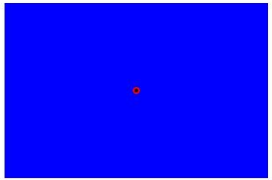
Vektorgrafik: SVG

Transformationen (Translation, Rotation):

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <rect x="50" y="100" width="200" height="300" fill="blue"
  transform="translate(50,  50)"/>
  <circle cx="200" cy="300" r="3" stroke-width="2"
   stroke="red"/>
</svg>
```



```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <rect x="100" y="200" width="200" height="300" fill="blue"
     transform="rotate(100,200,350)"/>
   <circle cx="200" cy="350" r="3" stroke-width="2"
```

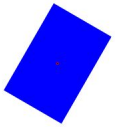


Vektorgrafik: SVG

Transformationen (Skalierung, Verkettung):

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <rect x="100" y="200" width="200" height="300" fill="blue"
     transform="scale(0.3)"/>
   <circle cx="200" cy="350" r="3" stroke-width="2" stroke="red"/>
</svg>
```

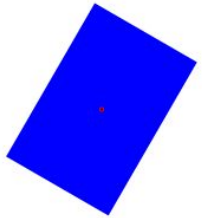
```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <rect x="100" y="200" width="200" height="300" fill="blue"
transform="translate(50,50), rotate(30,200,350),scale(0.5)"/>
   <circle cx="200" cy="350" r="3" stroke-width="2" stroke="red"
```



Vektorgrafik: SVG

Gruppierung <g> - Befehle werden auf alle Elemente der Gruppe angewendet:

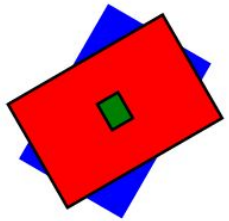
```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
    <g transform="translate(50,50), rotate(30,200,350),scale(0.5)">
        <rect x="100" y="200" width="200" height="300" fill="blue" />
        <circle cx="200" cy="350" r="3" stroke-width="2
    </g>
</svg>
```



Vektorgrafik: SVG

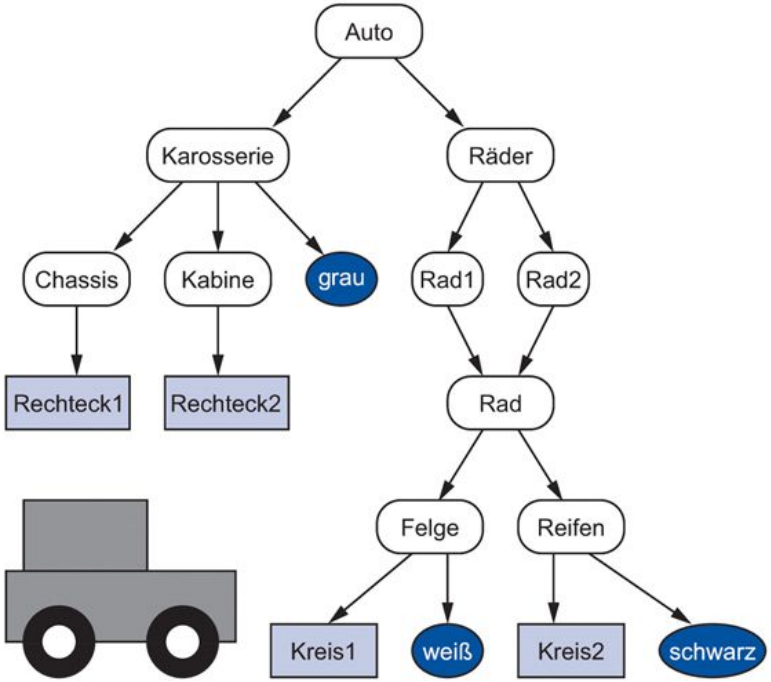
Gruppierung <g> - Gruppen können weitere Gruppen beinhalten:

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
<g transform="translate(50,50), rotate(30,200,350),scale(0.5)">
  <rect x="100" y="200" width="200" height="300" fill="blue" />
   <circle cx="200" cy="350" r="3" stroke-width="2" stroke="red"/>
   <g transform="rotate(30,200,350)" stroke="black" stroke-width  "5"
     <rect x="100" y="200" width="200" height="300" fill="red" />
     <rect x="100" y="200" width="50" height="40" fill="green"
       transform="translate(75,130)"/>
   </g>
</g>
</svg>
```



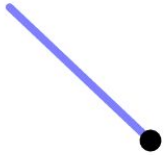
Szenengraphen: 2D und 3D Grafik

Szenengraph
für ein Auto



Vektorgrafik: SVG

SVG unterstützt auch Animationen



animateTransform-Element:

```
<svg xmlns="http://www.w3.org/2000/svg" width="300" height="500">
    <g transform="translate(70 50)" id="hour">
        <line stroke-width="5" x1="0" y1="0" x2="44" y2="44"
            stroke-linecap="round" stroke="blue" opacity=".5" />
        <animateTransform attributeName="transform" type="rotate"
            repeatCount="indefinite" dur="60s" by="360" />
        <circle r="7" />
    </g>
</svg>
```

Vektorgrafik

Kurven werden häufig durch *Splines* beschrieben.

- Bei GIMP und SVG z.B. durch *Bézier-Splines*

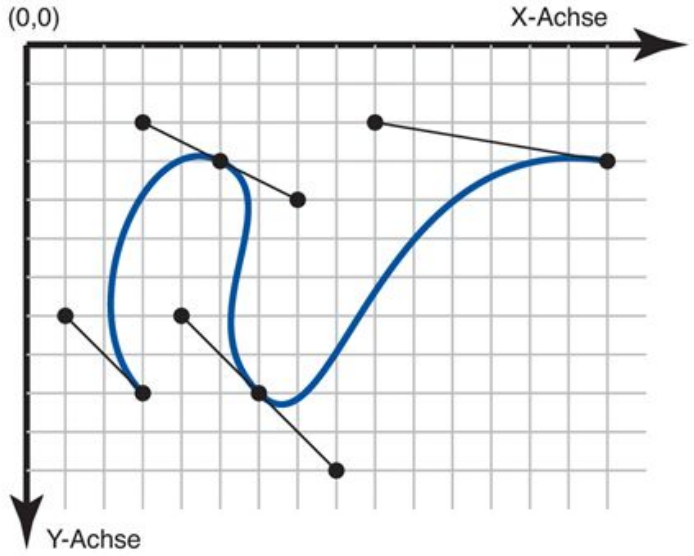


Abbildung aus: *Rainer Malaka, Andreas Butz, Heinrich Hußmann: Medieninformatik - Eine*

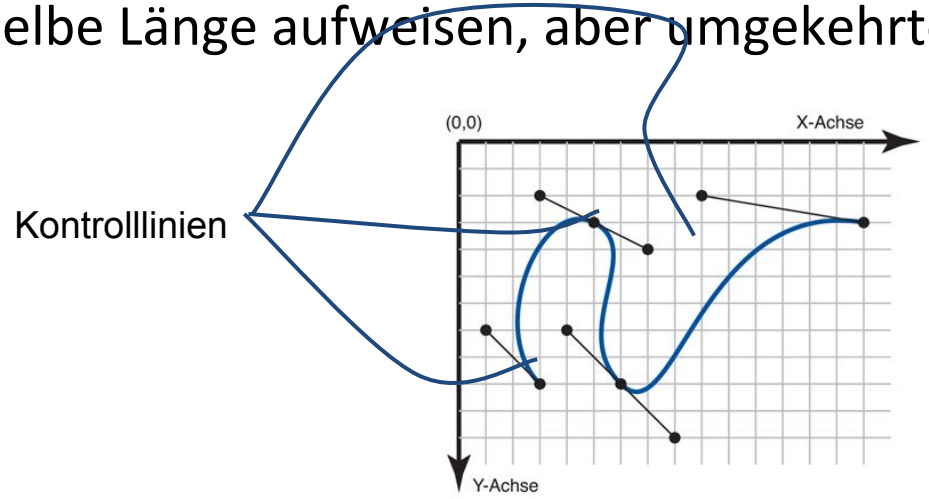
Vektorgrafik: Splines

Die Abbildung zeigt Bézier-Kurven ersten Grades, zweiten Grades (quadratisch), und dritten Grades (kubisch). Beachten Sie, dass die Anzahl der Kontrollpunk um eines höher ist als der Grad der Kurve.

Abbildung aus: *Rainer Malaka, Andreas Butz, Heinrich Hußmann: Medieninformatik - Eine*

Vektorgrafik: Splines

- Bei Bézier-Splines gelingt der Übergang von einem Teilstück zum nächsten "glatt" (die Krümmungen und Tangenten stimmen überein), wenn die Kontrolllinien der Teilstücke parallel verlaufen und dieselbe Länge aufweisen, aber umgekehrte Vorzeichen haben.



Vektorgrafik: Splines

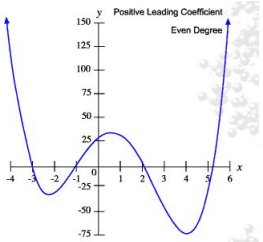
- Bei Bézier-Splines gelingt der Übergang von einem Teilstück zum nächsten "glatt" (die Krümmungen und Tangenten stimmen überein), wenn die Kontrolllinien der Teilstücke parallel verlaufen und dieselbe Länge aufweisen, aber umgekehrte Vorzeichen haben.

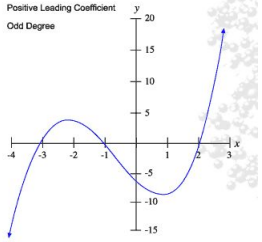
→ Natürlich müssen jeweils der Endstützpunkt und der Anfangssützpunkt der beiden kurven übereinstimmen.

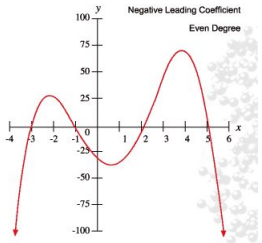
→ Wenn Forderung erfüllt ist, dann handelt es sich um einen “parametrisch stetigen Übergang”.

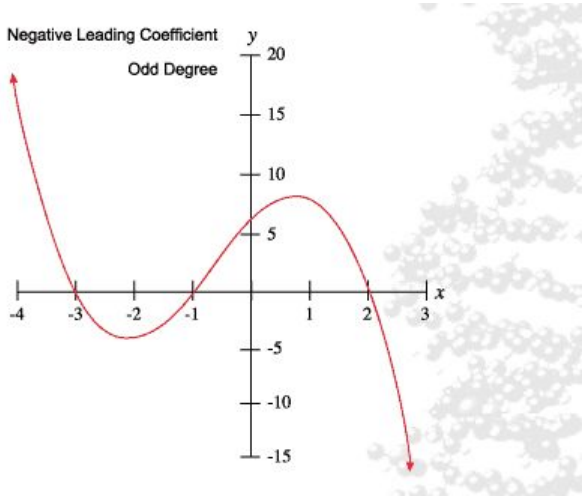
→ Optisch reicht jedoch auch ein “geometrisch stetiger” Übergang

Splines









http://www.biology.arizona.edu/biomath/tutorials/polynomial/Polynomialbasics.html

Splines

- Allgemein lassen sich Kurven *parametrisch* bestimmen.
  - Alternative Bestimmungsweisen: Implizit, funktional
- Beispiele:

$$x(t) = \cos(2\pi t),$$
$$y(t) = \sin(2\pi t).$$

mit $t \in [0,1]$

parametrisch

$$x^2 + y^2 = 1.$$

implizit

$$y = \sqrt{1 - x^2}.$$

explizit

Splines

- Allgemeine Form einer parametrischen Funktionsdarstellung:

$$\boldsymbol{P}(t) = (x(t), y(t))$$

Splines

- Beschreibung einer Kurve in parametrische Form mit Hilfe eines kubischen Polynoms:

$$x(t) = a_x t^3 + b_x t^2 + c_x t + d_x \qquad \text{mit } t \in [0,1]$$

$$y(t) = a_y t^3 + b_y t^2 + c_y t + d_y$$

- Das Polynom ist hier in seiner *Standardform* aufgeschrieben.
- Splines werden gebildet durch geschicktes aneinanderfügen von Kurvensegmenten, die durch Polynome beschrieben

Splines

- Die Punkte, an denen zwei polynomiale Kurvensegmente eines Splines aneinander stoßen, heißen *Knoten*.
- Ort und Form eines polynomialen Kurvensegments werden mit Hilfe von *Kontrollpunkten* beschrieben.
  - Die Kontrollpunkte kann man *nicht* unmittelbar an der Standardform ablesen; dafür sind spezielle Darstellungsformen notwendig.
- Bei grafischen Anwendungen setzt man häufig *kubische Polynome* ein.
  - Bei kubischen Polynomen gibt es einen guten Kompromiss zwischen Berechnungsaufwand und einem "gleichmäßigen" Verlauf der Kurve.

Splines

- *Bézier-Splines* sind eine Art und Weise, wie polynomiale parametrische Kurven zu Splines zusammengefügt werden können.
- Wir betrachten nur *kubische* Bézier-Splines.
- Kubische polynomiale parametrische Kurven haben vier Kontrollpunkte.
- Folgende parametrische Darstellung eines kubischen Béziers enthält explizit die vier Kontrollpunkte. Diese Darstellung verwendet eine *Bernstein-Basis*:

$$P(t) = (1-t)^3P_0+3t(1-t)^2P_1+3t^2(1-t)P_2+t^3P_3$$

  - Achtung: Es handelt sich weiterhin um ein Polynom, nur dass eine günstigere Darstellung für die Berechnung gewählt wurde

Splines

- Ein Punkt einer Bézier-Kurve zu einem gegebenen Parameter t läßt sich mit dem Algorithmus von de Casteljau schnell bestimmen:
  http://www.malinc.se/m/DeCasteljauAndBezier.php

Splines

Einige Eigenschaften von Bézier-Kurven

→ Die Kurve verläuft durch zwei der Kontrollpunkte - Start- und Endpunkt

→ Die Koeffizienten $P_{0, \ldots,} P_n$ besitzen eine geometrische Bedeutung.

→ Die Kurve ist *affin invariant*:

→ Verändern sich die Kontrollpunkte durch eine affine Transformation (z.B. Skalierung, Rotation, Translation), dann verschiebt sich die Kurve so, dass die entsprechend verschobenen Punkte der Ursprungskurve wieder auf der neuen Kurve liegen.

→ Die Punkte $P(t)$ der Kurve sind einfach zu berechnen.

→ Mit dem Algorithmus von de Casteljau liegen Verfahren vor, um Kurven polygonal zu approximieren und zu teilen.

Exkurs: Vektorrechnung

Diese Seite gibt eine gute Wiederholung der Grundbegriffe: http://nibis.ni.schule.de/~lbs-gym/Vektorpdf/EinfuehrungVR.pdf

Diese Seite illustriert die Zusammenhänge interaktiv:

http://phet.colorado.edu/sims/vector-addition/vector-addition_en.htm

Gegeben seien Vektor **a**=<3,1> und **b**=<1,2>.

- Wie berechnet man die Summe **a**+**b**?

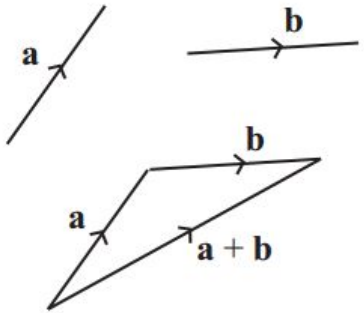
Exkurs: Vektorrechnung

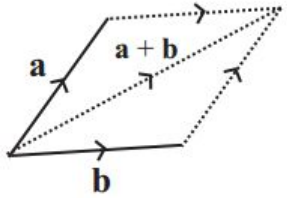
- Wie berechnet man die Summe **a**+**b** = <3+1,1+2 >= <4, 3>

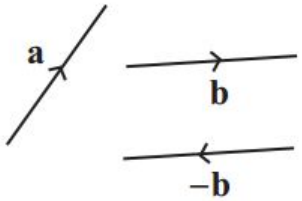
Vektorrechnung

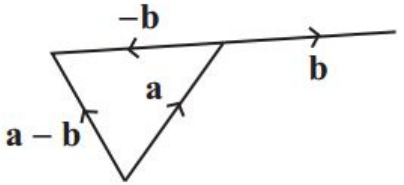
Cave! “Vektor”-Grafik besitzt nur indirekte und historische Zusammenhänge zu “Vektor”-Rechnung.

I





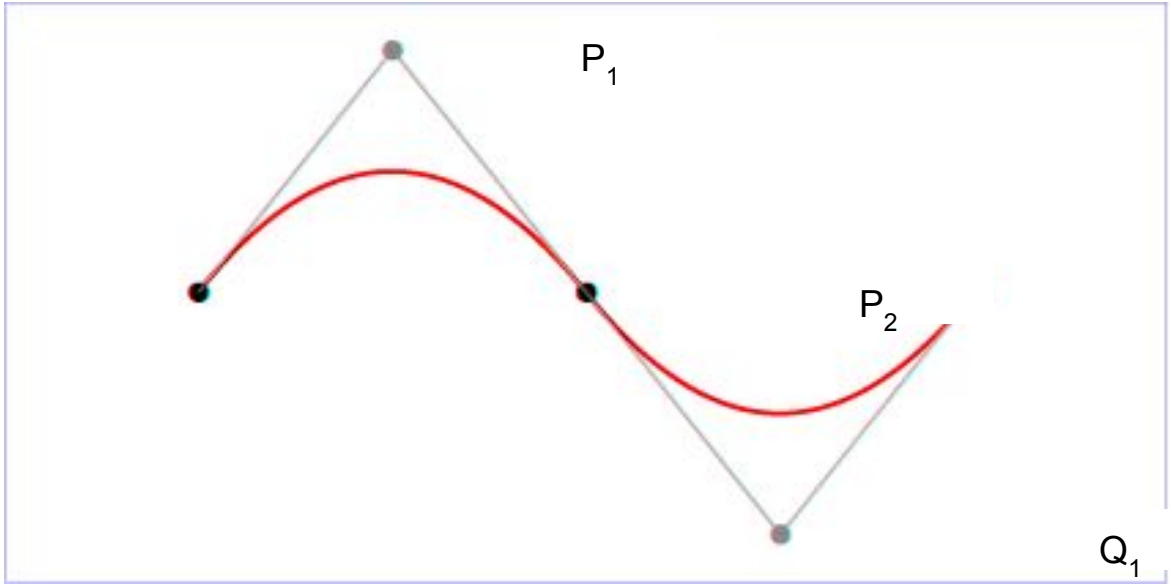




http://www.mathcentre.ac.uk/resources/uploaded/mc-ty-introvector-2009-1.pdf

Vektorgrafik: SVG

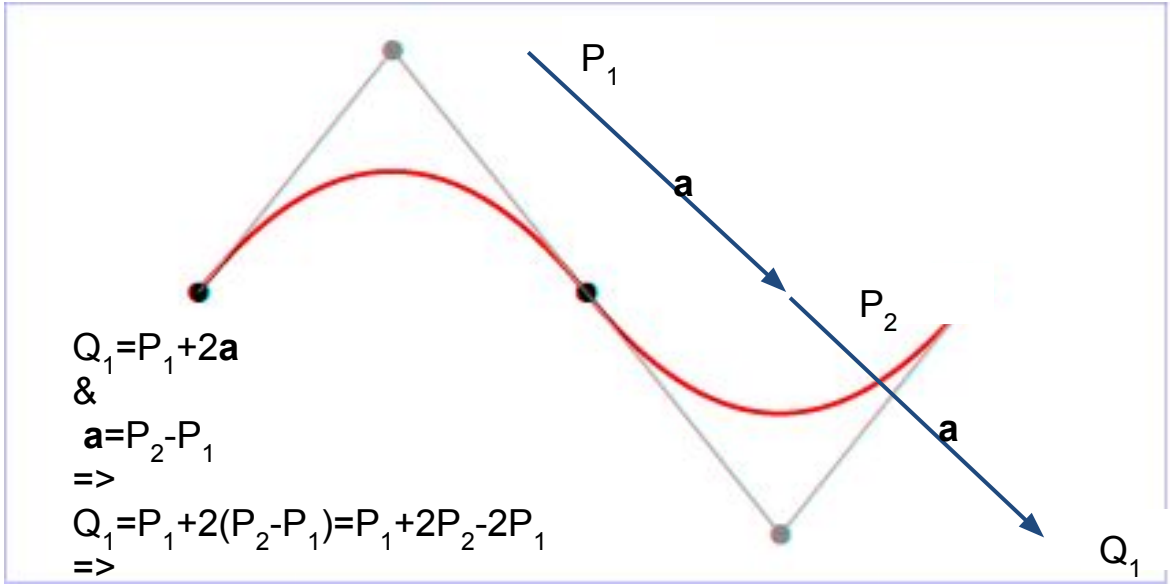
Wie wendet man Vektorrechnung an, um den Kontrollpunkt Q_1 zu berechnen, wenn P_1 und P_2 gegeben sind bei dieser quadratischen Bezier-Kurve?



http://www.w3.org/TR/SVG/paths.html#PathDataQuadraticBezierCommands

Vektorgrafik: SVG

Wie wendet man Vektorrechnung an, um den Kontrollpunkt Q_1 zu berechnen, wenn P_1 und P_2 gegeben sind bei dieser Quadratischen Bezier-Kurve?



$Q_1=P_1+2\mathbf{a}$
&
$\mathbf{a}=P_2-P_1$
=>
$Q_1=P_1+2(P_2-P_1)=P_1+2P_2-2P_1$
=>
$Q_1=2P_2-P_1$

http://www.w3.org/TR/SVG/paths.html#PathDataQuadraticBezierCommands

Vektorgrafik: SVG

Beispiel: Path-Anweisungen (Auswahl)

- M = moveto
- L = lineto
- C = curveto (cubic Bézier)
- S = smooth curveto (cubic Bézier)
- Q = curveto(quadratic Bézier)
- T = smooth curveto (quadratic Bézier)
- A = elliptical Arc
- Z = closepath

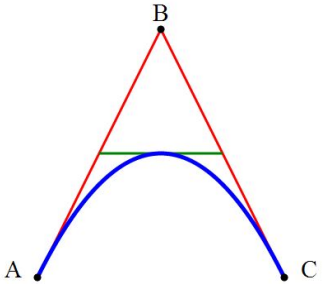
(Kleinbuchstaben - m, l, c, s, q, t a - bedeuten relative Positionen)

Beispiele aus http://www.w3schools.com/

Vektorgrafik: SVG

Beispiel: Path-Anweisungen (Auswahl)

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <path id="lineAB" d="M 100 350 l 150 -300" stroke="red"
  stroke-width="3" fill="none" />
  <path id="lineBC" d="M 250 50 l 150 300" stroke="red"
  stroke-width="3" fill="none" />
  <path d="M 175 200 l 150 0" stroke="green" stroke-wid
  fill="none" />
  <path d="M 100 350 q 150 -300 300 0" stroke="blue"
  stroke-width="5" fill="none" />
  <!-- Mark relevant points -->
  <g stroke="black" stroke-width="3" fill="black">
    <circle id="pointA" cx="100" cy="350" r="3" />
    <circle id="pointB" cx="250" cy="50" r="3" />
    <circle id="pointC" cx="400" cy="350" r="3" />
  </g>
  <!-- Label the points -->
  <g font-size="30" font="sans-serif" fill="black" stroke="none"
```



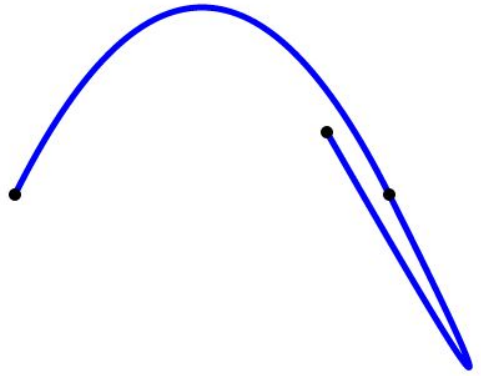
http://www.w3schools.com/svg/svg_path.asp

Beispiele aus http://www.w3schools.com/

Vektorgrafik: SVG

Path-Anweisungen: Forführung eines Bézier-Teilstücks zu einem glatten Bézier-Spline mit dem Befehl *t*.

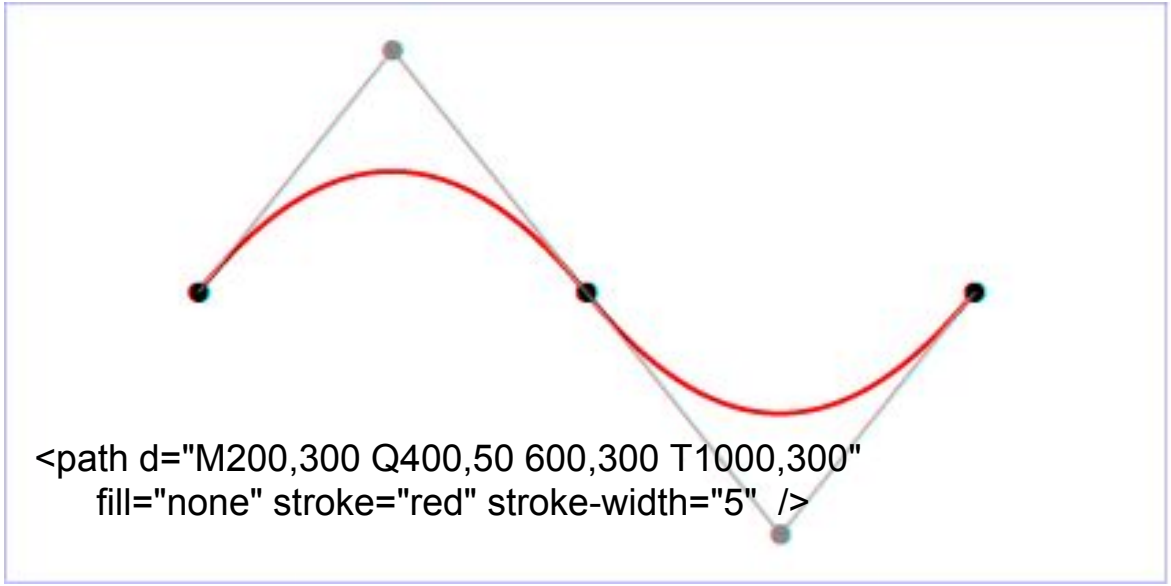
```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">

  <path d="M 100 350 q 300 0 150 -300 t -50 -50 "
stroke="blue"
  stroke-width="5" fill="none" />

 <circle cx="100"  cy="350" r="5"/>
 <circle cx="250"  cy="50" r="5"/>
 <circle cx="400"  cy="350" r="5"/>
 <circle cx="350"  cy="300" r="5"/>
```



Beispiele aus http://www.w3schools.com/

Vektorgrafik: SVG

Path-Anweisungen: Forführung eines Bézier-Teilstücks zu einem glatten Bézier-Spline mit dem Befehl *T*.

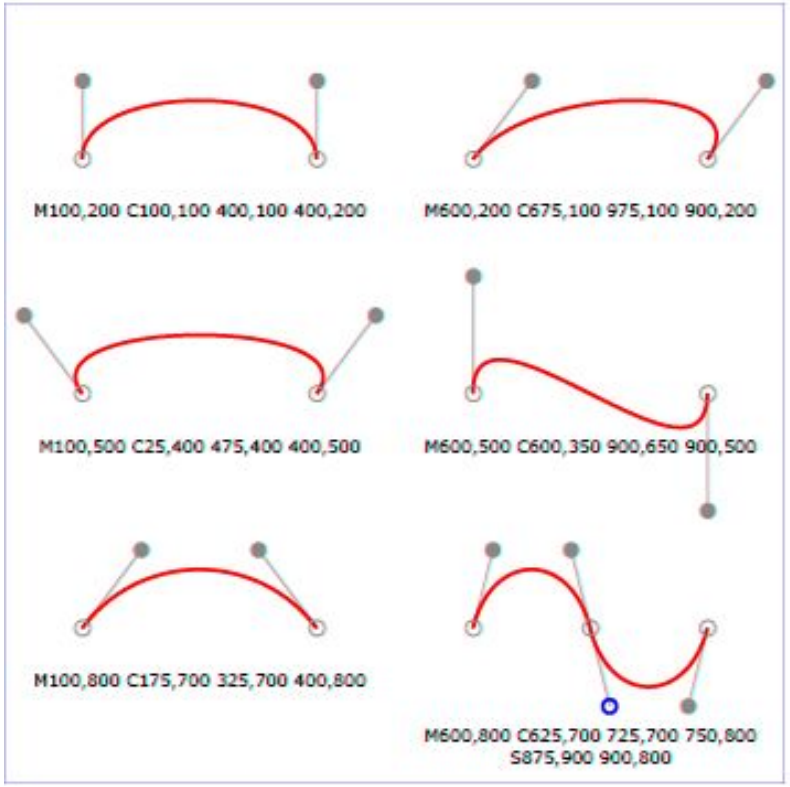


```
<path d="M200,300 Q400,50 600,300 T1000,300"
      fill="none" stroke="red" stroke-width="5"  />
```

http://www.w3.org/TR/SVG/paths.html#PathDataQuadraticBezierCommands

Vektorgrafik: SVG

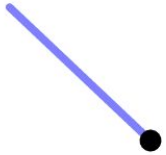
Path-Anweisungen: Bézier-Splines



http://www.w3.org/TR/SVG/paths.html#PathDataQuadraticBezierCommands

Vektorgrafik: SVG

SVG unterstützt auch Animationen



animateTransform-Element:

```
<svg xmlns="http://www.w3.org/2000/svg" width="300" height="500">
    <g transform="translate(70 50)" id="hour">
        <line stroke-width="5" x1="0" y1="0" x2="44" y2="44"
            stroke-linecap="round" stroke="blue" opacity=".5" />
        <animateTransform attributeName="transform" type="rotate"
            repeatCount="indefinite" dur="60s" by="360" />
        <circle r="7" />
    </g>
</svg>
```

Vektorgrafik: SVG

Animation entlang eines Pfades: *animateMotion*-Element mit kubischer Bézier-Kurve

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<svg xmlns="http://www.w3.org/2000/svg">
    <path stroke="black" d="M20 100 C20 0 180 0 180 100"
        stroke-width="2" fill="none"/>
    <rect x="-5" y="-5" width="10" height="10" fill="blue">
        <animateMotion path="M20 100 C20 0 180 0 180 100" dur="10s"
            rotate="auto" repeatCount="3" fill="freeze"/>
    </rect>
</svg>
```