EduFab – Workshop : „Programmierte Schmuckstücke”

Ein 3-stündiger Schnupperworkshop im FabLab

AG Digitale Medien in der Bildung

Nadine Dittert, Dennis Krannich, Bernard Robben

Kontakt: nditterT@tzi.de

15. Januar 2016

eduFab

Dieses Dokument ist im Rahmen des BMBF-geförderten Projekts EduFab an der Universität Bremen entstanden. EduFab begann im November 2013 und endet im Juli 2016.

Universität Bremen*

dimeb

Make Light
Photonik selber machen

EINE INITIATIVE VOM
Bundesministerium
für Bildung
und Forschung

Inhaltsverzeichnis

INHALTSVERZEICHNIS 3

1. KURZBESCHREIBUNG 4

2. INHALT DES WORKSHOPS: PROGRAMMIERUNG MIT PROCESSING 4

2.1. Funktionsweise eines Laser-Cutters 4

2.2. Programmieren mit Processing 4

3. VORBEREITUNG 5

3.1. Bereitstellen des Konstruktionsmaterials 5

3.2. Vorbereiten der Arbeitsplätze 5

4. ABLAUF DES WORKSHOPS 6

4.1. Begrüßung 6

4.2. Vorstellungsrunde 6

4.3. Wie funktioniert ein Laser-Cutter und was ist das? 6

4.4. Programmiereinführung 6

4.6. Präsentation 8

4.7. Abschluss: Bezug zum Alltag 8

ANHANG A: EINKAUFSLISTE 10

ANHANG B: HANDOUTS ZUR PROGRAMMIERUNG MIT PROCESSING 11

1. Kurzbeschreibung

„**Programmierte Schmuckstücke**" ist ein Kurzworkshop, der Anfängerinnen und Anfängern einen Einblick in FabLab-Technologien und Programmierung mit Processing gibt. In drei Stunden programmieren die Teilnehmenden ein Muster, das sie mit dem Laser-Cutter ausschneiden und wenn gewünscht gravieren können, um es dann als Kettenanhänger oder Ohrringe zu tragen. Dieses Format wurde am Girls' Day mit 11 Mädchen der Klassenstufen 7 und 8 mit drei Betreuerinnen erprobt. In einer erweiterten Version wurde es in einem 2-tägigen Workshop erfolgreich durchgeführt.

2. Inhalt des Workshops: Programmierung mit Processing

In diesem Workshop werden zwei Bereiche thematisiert: die Funktionsweise von Laserschneidegeräten (Laser-Cutter) und Programmierung (mit Processing).

2.1. Funktionsweise eines Laser-Cutters

In diesem Workshop werden die grundlegenden Begriffe Laser und Laser-Cutter erklärt. Dabei wird darauf eingegangen, was Licht ist und wie es entsteht. Licht wird als schwingende Energieteilchen, die sich von einer Quelle aus wellenförmig ausbreiten, erklärt. Das Prinzip eines Lasers wird vereinfacht anhand einer Lupe erklärt, mit der Sonnenlichtstrahlen gebündelt werden können. Dabei entsteht an einem Punkt (dem Brennpunkt) sehr große Hitze und Brandgefahr. Beim Laser wird das Licht auf ähnliche Weise fokussiert und kann dadurch soviel Kraft an einer bestimmten Stelle entwickeln, dass damit Material geschnitten werden kann. Ähnlich wie beim Sonnenlicht sollte man nicht direkt in den Laser schauen.
Ein Laser-Cutter lässt sich ähnlich wie ein Drucker bedienen. Er erhält eine Datei mit Informationen, wo geschnitten oder graviert werden soll. Wichtig ist zudem die Fokussierung des Lasers auf das Werkstück.

2.2. Programmieren mit Processing

Processing[1] ist eine textuelle Programmiersprache, die mit einer einfachen Version von Java vergleichbar ist. Die Entwicklung von Processing begann im Jahr 2001 mit dem Hintergrund, Künstlerinnen und Künstlern Programmierung als Werkzeug zur Verfügung zu stellen. Processing hat vorwiegend grafische Anwendungen zum Ziel, ist jedoch für diverse andere Nutzungsmöglichkeiten offen. Es ist quelloffen und damit für alle, denen die technischen Möglichkeiten zur Verfügung stehen, frei zugänglich.

Processing arbeitet hauptsächlich in zwei Methoden - *setup()* und *draw()*, auf die wir in unserem Konzept aus folgenden Gründen verzichten: Wir möchten die Teilnehmenden nicht mit unnötigen Informationen „belasten" und wir möchten ihnen das Gefühl geben, dass sie ihr Programm komplett selbst gestalten und darüber „herrschen". Da Processing es ermöglicht, mit einem „leeren Blatt" zu beginnen, nutzen wir dies. Wir weisen an dieser Stelle ausdrücklich darauf hin, dass dieses Konzept auf Programmieranfängerinnen und -anfänger ausgerichtet ist.
Als Programmierwerkzeuge konzentrieren wir uns auf zwei verschiedene Möglichkeiten: Geraden zeichnen (als Linien, Rechtecke, Dreiecke oder Vierecke) und Kreise oder Teile davon zu zeichnen (als Kreise, Ellipsen oder Bogen). Mit diesen Werkzeugen ist es möglich, einfache

[1] http.//www.processing.org, letzter Zugriff am 10.06.2016

Muster und Figuren individueller Form zu gestalten. Für den gewählten Zeitrahmen mit Programmieranfängerinnen stellten sich diese Möglichkeiten als geeignet heraus[2].

Geraden lassen sich mit dem Befehl *line()* zeichnen, dem als Parameter x- und y-Werte eines Start- und eines Endpunktes übergeben werden. Ein Rechteck wird mit dem Befehl *rect()* erzeugt, dem als Parameter der linke, obere Eckpunkt sowie die Breite (nach rechts) und die Höhe (nach unten) übergeben werden. Bei Drei- und Vierecken werden den Befehlen *triangle()* und *quad()* die Koordinaten der drei bzw. vier Eckpunkte als Parameter übergeben. Beispiele dazu befinden sich im Handout, das den Teilnehmenden für die Programmierung zur Verfügung gestellt wird und in Anhang B zu sehen ist.
Kreise und Ellipsen werden mit dem Befehl *ellipse()* erzeugt. Als Parameter werden hierbei der (Mittel-) Punkt, um den der Kreis oder die Ellipse gezeichnet werden soll übergeben sowie die gesamte Breite und Höhe, die im Falle eines Kreises identisch sind (siehe Anhang B). Für Bogen wird der Befehl *arc()* mit den gleichen Parametern eingesetzt, die durch einen Anfangs- und Endpunkt (z.B. von 0 bis PI für einen Halbkreis) erweitert werden (siehe Anhang B).

3. Vorbereitung

3.1. Bereitstellen des Konstruktionsmaterials

Als Schmuckstücke sollen Ketten oder Ohrringe entstehen. Entsprechend können Lederbänder verschiedener Farben, Zwischenringe zur Anbringung der Anhänger sowie Ohrfedern oder Ohrstecker und Kleber zur Befestigung zur Verfügung gestellt werden. Als Werkzeug haben sich Schmuckzangen, mit denen sich die Zwischenringe verbiegen lassen, als hilfreich erwiesen.
Für die Anhänger selbst eignet sich Holz (MDF), sowie Plexiglas und Acryl mit verschiedenen Farben. Aus organisatorischen Gründen hat sich gezeigt, dass es hilfreich ist, mit einer einheitlichen Materialstärke (z.B. 3mm) zu arbeiten. So wird zum einen bei einigen Programmen u. a. verhindert, dass der Laser bei einem Materialwechsel neu fokussiert werden muss und zum anderen, dass die Laserwerte (Stärke und Geschwindigkeit) neu eingestellt und optimiert werden müssen. Diese beiden Vorgänge sind eher zeitintensiv und für den hier beschriebenen zeitlichen Rahmen ungeeignet. Eine Beispieleinkaufsliste befindet sich in Anhang A.

3.2. Vorbereiten der Arbeitsplätze

Für die Programmierung zu zweit wird jeweils ein Rechner bzw. Laptop benötigt, der bereits offen hingestellt werden kann. Die Programmierumgebung Processing (mit einer vorhandenen svg-Bibliothek) kann ebenso bereits gestartet sein. Die Handouts können zu Beginn oder auch erst nach der Programmiereinführung verteilt werden. Für die Skizzen, die die Teilnehmenden zur Kommunikation untereinander und für ihre Arbeit generell erstellen, kann kariertes oder Millimeterpapier mit Stiften zur Verfügung gestellt werden.

[2] Schleifen für wiederkehrende Muster wurden zusätzlich erklärt, jedoch von den Teilnehmerinnen nicht für ihre Muster benutzt.

In unserem Arbeitsumfeld, in dem der Laser-Cutter von einem zentralen Rechner aus angesteuert wird, haben sich USB-Sticks zur Übertragung der Dateien bewährt.

4. Ablauf des Workshops

4.1. Begrüßung

Anfangs werden die Teilnehmenden Willkommen geheißen. Die Tutorinnen und Tutoren stellen sich vor. Dabei ist für die Teilnehmenden häufig interessant, warum der Workshop stattfindet und insbesondere beim Girls' Day was die Tutorinnen und Tutoren sonst in ihrem Beruf machen.
Es wird erklärt, was ein FabLab ist und darin die Möglichkeit betont, dass dort jede und jeder Dinge entwerfen und herstellen kann. Es wird kurz auf den Inhalt des Workshops hingewiesen, in dem Programmierung als Herstellungsart genutzt wird.

4.2. Vorstellungsrunde

In Abhängigkeit vom Kontext macht eine Vorstellungsrunde den Workshop etwas persönlicher. Befinden wir uns im Schulkontext, in dem sich Teilnehmende und Tutorinnen bzw. Tutoren bereits kennen, ist dies nicht notwendig.

Die Teilnehmenden stellen sich namentlich vor. Für die Leitenden kann von Interesse sein, warum die Teilnehmenden im Workshop sind und welche Vorerfahrungen sie mitbringen, um entsprechend darauf eingehen zu können. Die Vorstellungsrunde dauert etwa fünf Minuten.

4.3. Fantasiephase

Zu Beginn sollen die Teilnehmenden ihre Ideen einbringen und ihrer Fantasie freien Lauf lassen. Mit Papier und Stift, Knete oder einfachen Formen aus Moosgummi können sie Muster oder Formen entwickeln, mit denen sie selbst etwas verbinden oder die sie mögen. Nach etwa 5 bis 10 Minuten stellen sie ihre Ideen vor.

4.4. Wie funktioniert ein Laser-Cutter und was ist das?

In diesem Teil findet eine kurze theoretische Einführung in den Laser-Cutter statt, die sich etwa über 10 bis 15 Minuten erstreckt.

Um die Teilnehmenden einzubinden, kann mit der Frage gestartet werden, wo im Alltag der Teilnehmenden Laser vorkommen. Mögliche Antworten sind Warenscanner, CD- oder DVD-Spieler, Laserpointer oder auch bei Augen-OPs verwendete Laserskalpelle. Danach wird besprochen, was Licht ist und wie ein Laser-Cutter damit arbeitet (siehe Abschnitt 2.1).

4.5. Programmiereinführung

Für die Programmiereinführung ist es hilfreich, parallel an einer Tafel (z.B. Whiteboard) und einem an einen Beamer angeschlossenen Rechner zu arbeiten. Grundlegend wird zunächst

geklärt, was Programmieren eigentlich ist. Das kann in der Runde gemeinsam erarbeitet werden. Ziel ist es, festzustellen, dass der Rechner selbst nichts tut außer dem, was ihm über Programmierung mitgeteilt wird. Dabei muss man sehr genau sein, da er es sonst nicht versteht. Begonnen wird mit einem leeren Processing-Programm, das einmal ausgeführt wird. Im Anschluss wird der Befehl *size()* erklärt, der das Fenster in einer bestimmten Größe (in unserem Fall hat sich *size(400, 400)* bewährt, siehe Zeile 9 in Quellcode1) zeichnet. Dies wird mit Hilfe eines Koordinatensystems am Whiteboard erläutert. Von der linken oberen Ecke ausgehend werden Koordinaten in dem Fenster erklärt. Es lässt sich mit dem 4. Quadranten eines aus der Schule bekannten Koordinatensystems vergleichen, wobei hier im Unterschied dazu beide Werte positiv sind (siehe Abbildung 1).

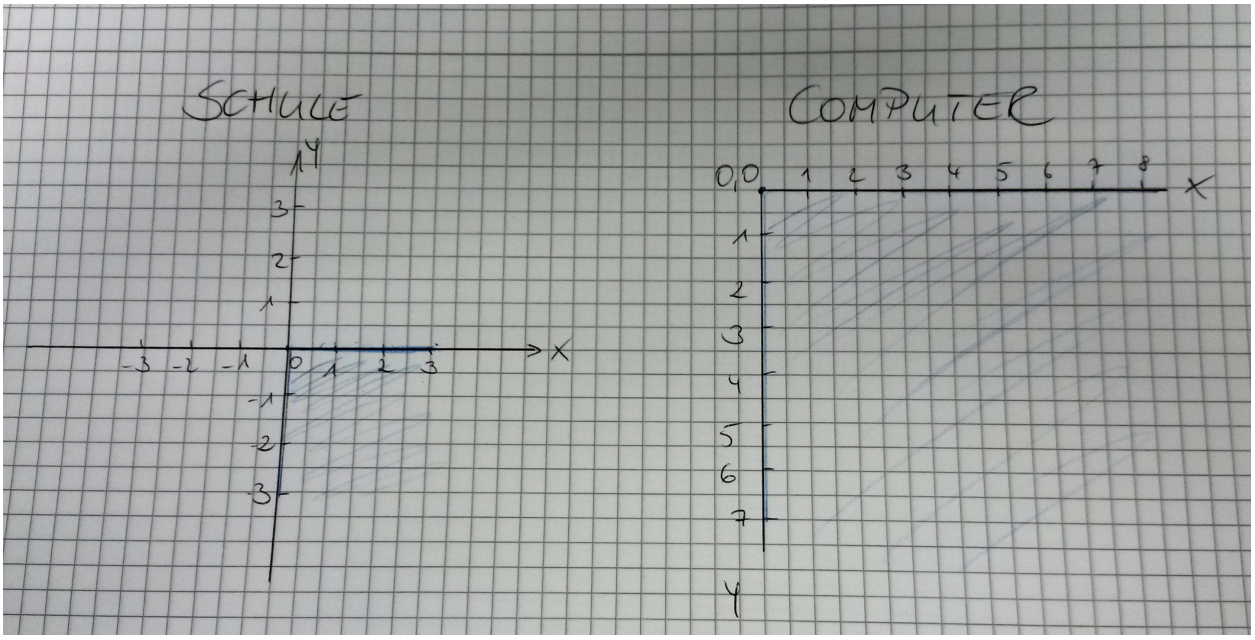


Abbildung 1: Vergleich der Aufteilung eines Koordinatensystems und des Processing-fensters

Der Hintergrund dieses Fensters wird mit dem Befehl *background(255)* (siehe Zeile 10 in Quellcode 1) weiß gezeichnet, wobei 255 den Farbwert Weiß angibt (und 0 den Farbwert Schwarz). Damit ist zunächst der Grundrahmen des Programms fertig und wir können mit den eigentlichen Mustern fortfahren. Wir beginnen mit Linien, die von einem Punkt A zu einem Punkt B gezeichnet werden. Um eine Linie vom Punkt (30, 20) zum Punkt (85, 75) zu zeichnen schreiben wir *line(30,20,85,75);* (siehe Zeile 13 in Quellcode 1). Dies zeigen wir einmal zeichnerisch am Whiteboard und im Anschluss am Rechner. Dort wird dann auch das Programm gestartet und damit gezeigt, dass es genau dies tut. Um die Teilnehmenden wieder möglichst viel einzubinden, kann eine Frage - beispielsweise nach einer zweiten Linie zum Punkt (110, 50) zu zeichnen - gestellt werden. Es folgt der Befehl *line(85,75, 110,50);* (siehe Zeile 14 in Quellcode 1). Dieser wird ebenso zunächst am Whiteboard notiert und veranschaulicht, und im Anschluss am Rechner ausgeführt.

```
1  import org.philhosoft.p8g.svg.*;

3  P8gGraphicsSVG svg;
4  String svg_dateiname = "muster.svg";
5  svg = (P8gGraphicsSVG) createGraphics(width, height, P8gGraphicsSVG.SVG,
6                    svg_dateiname);
7  beginRecord(svg);

9  size(400,400);
10 background(255);


13 line(30,20,85,75);
14 line(85,75, 110,50);

16 svg.endRecord(); // hiermit beenden wir die SVG-Aufnahme
```

Quellcode 1: Beispielprogramm in Processing

Um nun wieder die Teilnehmenden aktiv einzubinden, sollen sie ein erstes eigenes Programm schreiben, in dem sich zwei Linien irgendwo kreuzen. Auf diese Weise findet die erste aktive Programmiertätigkeit statt und die Teilnehmenden haben ihr erstes Erfolgserlebnis. Unterstützt werden sie dabei von den Tutorinnen und Tutoren, die bei Fragen und Problemen bereit stehen. Auf Dreiecke, Vierecke, Kreise und Bogen wird nicht mehr in der großen Gruppe eingegangen, sondern es wird nur auf die Möglichkeit und die Handouts hingewiesen. Die Tutorinnen und Tutoren unterstützen direkt in den Kleingruppen.
Bevor die entstandene Grafik zum Laser-Cutter geschickt werden kann, muss die Ausgabe in eine SVG-Datei erfolgen. Hierfür muss zunächst in Zeile 1 ein die entsprechende Bibliothek eingebunden werden und ein svg-Objekt sowie eine Datei erstellt werden (Zeile 3-6). In Zeile 7 erfolgt der Befehl der die Ausgabe startet und in Zeile 16 der Befehl zum Beenden der Ausgabe.

4.6. Eigene Programmierung

Nach der Programmiereinführung können die Teilnehmenden in ihren Zweiergruppen arbeiten. Sie können sich an ihren Ideen aus der Fantasiephase orientieren, um eine Form, eine Figur oder ein Muster zu überlegen und es programmieren. Alternativ besteht die Möglichkeit, durch experimentelle Programmierung Muster zu entwerfen. Dabei werden sie von den Tutorinnen und Tutoren unterstützt, sofern dies erforderlich ist. Je nach Anzahl der Teilnehmenden ist darauf zu achten, dass ausreichend Zeit bleibt, die Formen am Ende zu gravieren und zu schneiden.

4.7. Präsentation

Am Ende präsentieren die Teilnehmenden ihre Schmuckstücke gegenseitig. Sie erhalten dabei Anerkennung für ihre Arbeit seitens der anderen Teilnehmenden sowie der Tutorinnen und Tutoren.

4.8. Abschluss: Bezug zum Alltag

Am Ende kann kurz darauf eingegangen werden, an welchen Stellen im (beruflichen) Alltag Dinge wie Programmierung und Laser-Cutter eingesetzt werden. Dabei sind die Aspekte der

Reproduzierbarkeit und der Erweiterbarkeit nennenswert. Außerdem lassen sich Modelle schnell prototypisch umsetzen, die dann mit Hilfe größerer Geräte (z.B. Fräsen) auf größere Gegenstände übertragen werden können. Ein Beispiel hierfür ist der im FabLab Gronigen (NL) entstandene „stool" in Abbildung 2. Weitere für Anwendungen lassen sich auf YouTube finden, beispielsweise aus der industriellen Produktion[3].

Abbildung 2: Laser-Cut-Modell und Möbelstück eines „stools"

[3] http://www.youtube.com/watch?v=umJcrGn4buM, letzter Zugriff am 07.01.2016

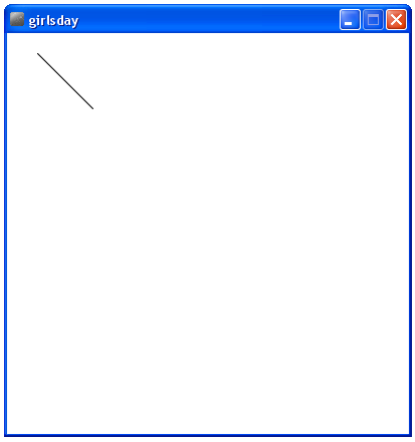
Anhang A: Einkaufsliste

Die hier aufgelisteten Dinge entsprechen den Vorschlägen aus Abschnitt 3.1, welches Material für den Workshop bereitgestellt werden kann.

Bezeichnung	Wo ist es erhältlich?
Lederbänder	Bastelladen, z.B. Opitec
Zwischenringe	Bastelladen, z.B. Opitec
Ohrfedern	Bastelladen, z.B. Opitec
Ohrstecker	Bastelladen, z.B. Opitec
Schmuckkleber	Bastelladen, z.B. Opitec
Schmuckzangen	Bastelladen, z.B. Opitec
MDF-Platten	Baumarkt, z.B. Hornbach, Obi
Transparentes Acrylglas, z.B. Plexiglas	Fachhandel, z.B. Innograv oder Gerstaecker
zweischichtiges Acryl, z. B. LaserAcryl oder LaserFlex	im Fachhandel, z.B. Innograv
Millimeterpapier	Schreibwarenladen
Bleistifte	Schreibwarenladen
USB-Sticks	Elektronikfachgeschäft

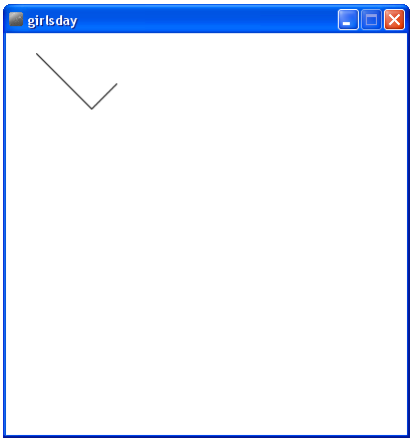
Tabelle1: Einkaufsliste für den Workshop „Programmierte Schmuckstücke“

1. Linien, Dreiecke und Vierecke



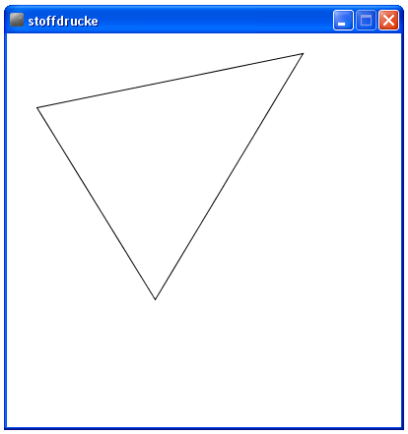

```
line(30, 20, 85, 75);
```

zeichnet eine **Linie** (engl. *line*) von
Punkt (30, 20) = Starpunkt der Linie
zu Punkt (85, 75) = Endpunkt der Linie



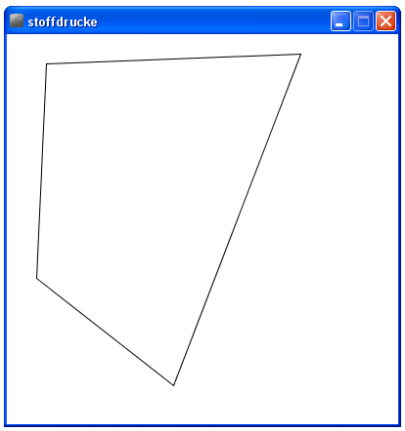

```
line(30, 20, 85, 75);
line(85, 75, 110, 50);
```

zeichnet eine weitere **Linie** von
Punkt (85, 75) = Endpunkt der ersten Linie
und gleichzeitig Startpunkt der zweiten Linie
zu Punkt (110, 50) = Endpunkt der zweiten Linie




```
triangle(30, 75, 300, 20, 150, 270);
```

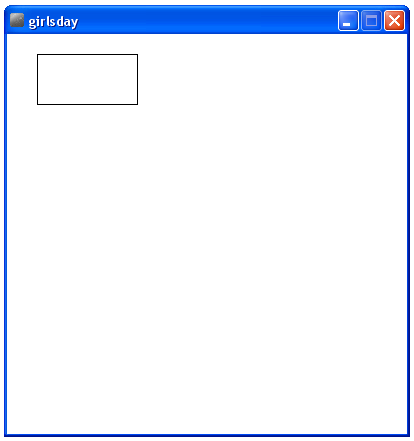
zeichnet ein geschlossenes **Dreieck**
(engl. *triangle*) von
Punkt (30, 75) = erster Eckpunkt
über Punkt (300, 20) = zweiter Eckpunkt
über Punkt (150, 270) = dritter Eckpunkt
zurück zum ersten Eckpunkt




```
quad(40, 30, 300, 20, 170, 360, 30, 250);
```

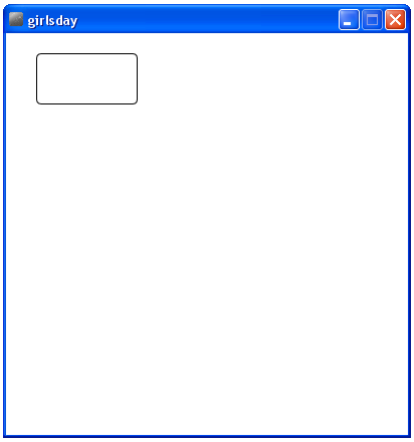
zeichnet ein geschlossenes **Viereck**
(engl. *quadrilateral*) von
Punkt (40, 30) = erster Eckpunkt
über Punkt (300,20) = zweiter Eckpunkt
über Punkt (170, 360) = dritter Eckpunkt
über Punkt (30, 250) = vierter Eckpunkt
zurück zum ersten Eckpunkt

2. Rechtecke




```
rect(30, 20, 100, 50);
```

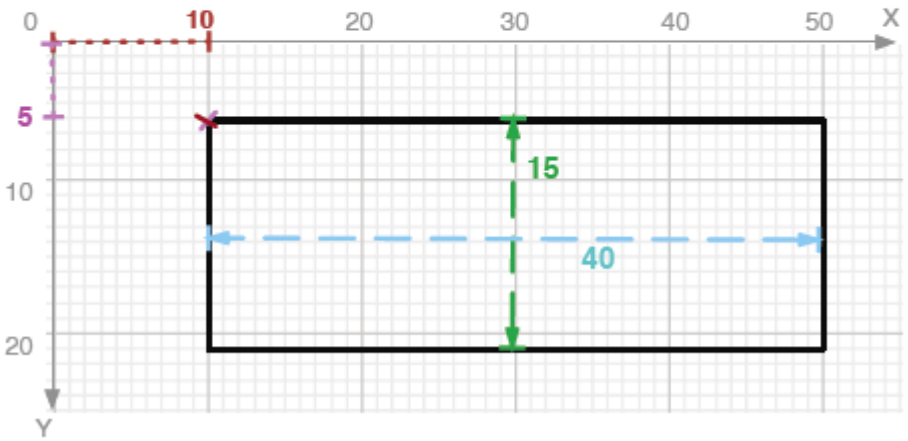
zeichnet ein Rechteck (engl. *rectangle*) von Punkt (30, 20) = linker oberer Eckpunkt
mit einer Breite von 100 Einheiten (nach rechts) und
einer Höhe von 50 Einheiten (nach unten)



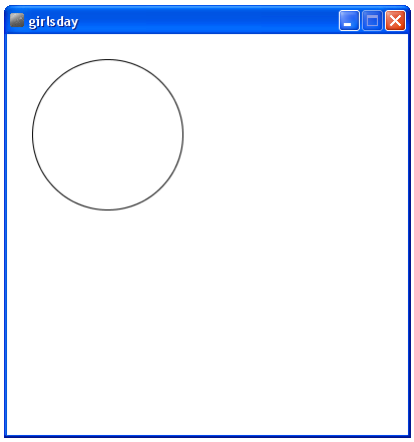

```
rect(30, 20, 100, 50,5);
```

zeichnet ein Rechteck mit „runden“ Ecken

```
rect( 10, 5, 40, 15 );
rect(  x, y,  b,  h  );
```

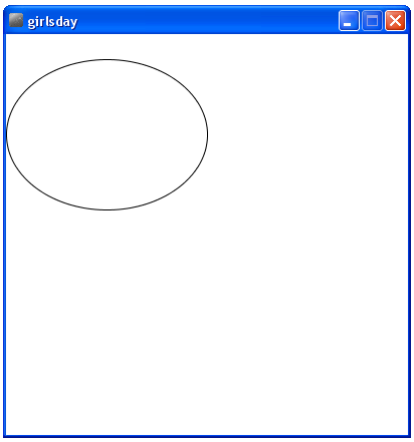


3. Kreise, Ellipsen und Bogen




```
ellipse(100, 100, 150, 150);
```

zeichnet einen Kreis (eine bestimmte Form einer Ellipse) um den Mittelpunkt (100, 100)
mit gleicher Breite und Höhe von 150 Einheiten (jeweils 75 Einheiten nach oben, nach unten, nach links und nach rechts)




```
ellipse(100, 100, 200, 150);
```

zeichnet eine Ellipse (engl. *ellipse*) um den (Mittel-) Punkt (100, 100)
mit einer Breite von 200 Einheiten (100 nach links und 100 nach rechts) und
einer Höhe von 150 Einheiten (75 nach unten und 75 nach oben)


```
arc(100, 100, 150, 150, 0, PI);
```

zeichnet einen Bogen (engl. *arc*) um den Punkt (100, 100)
mit gleicher Breite und Höhe von 150 Einheiten beginnend beim Punkt 0 auf dem Kreis (rechter oberer Punkt des Kreises) einen Halbkreis (=PI) lang

ellipse(30, 15, 40, 20);

ellipse(x, y, b, h);