EduFab – Workshop :: „Stoffdrucke programmieren”

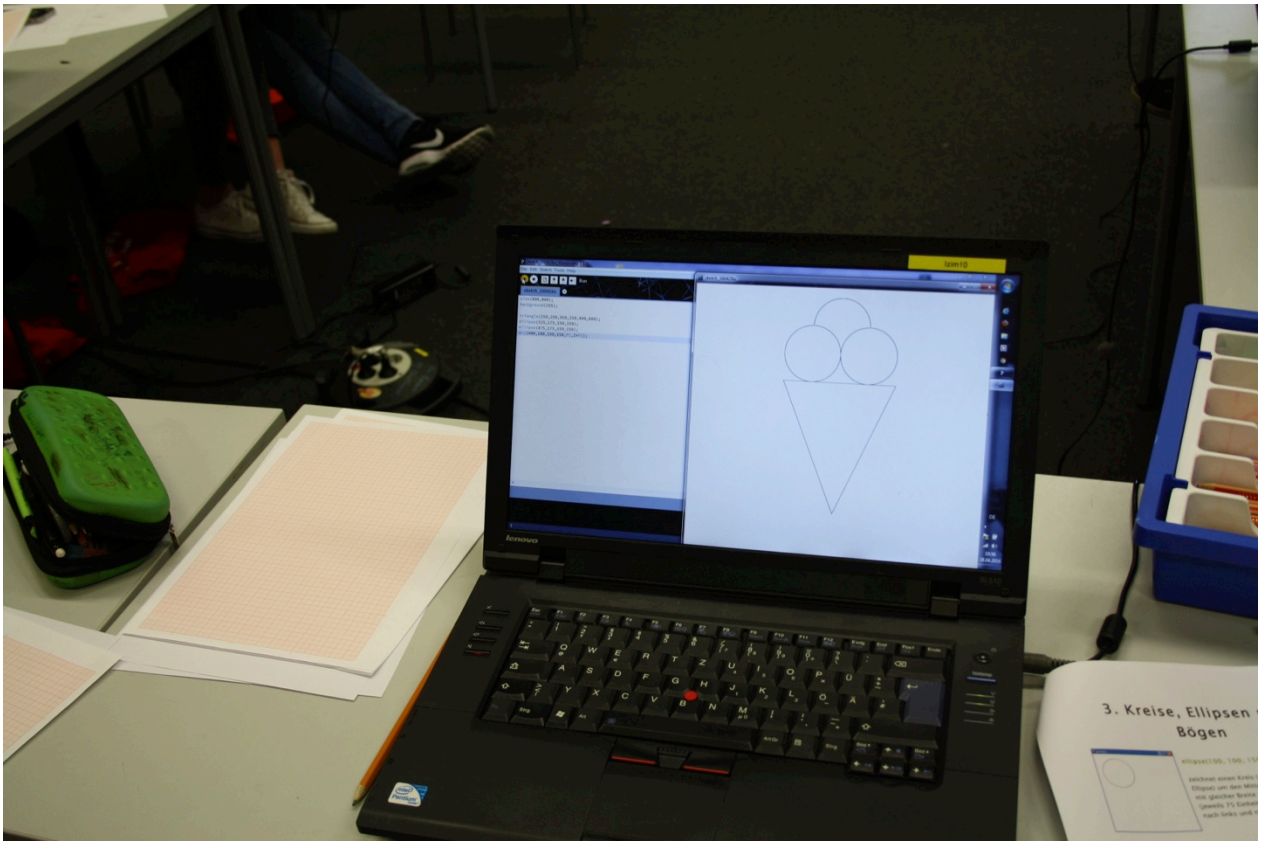
Ein 3-stündiger Schnupperworkshop im FabLab

Universität Bremen, AG Digitale Medien in der Bildung

Nadine Dittert, Bernard Robben

Kontakt: nditterT@tzi.de

08. September 2016



eduFab

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

Universität Bremen*

dimeb

Make Light
Photonik selber machen

EINE INITIATIVE VOM
Bundesministerium
für Bildung
und Forschung

Inhaltsverzeichnis

1. Kurzbeschreibung

„**Stoffdrucke programmieren**" ist ein Kurzworkshop, der Anfängerinnen und Anfängern einen praktischen Einstieg in textuelle Programmierung mit Processing gibt und dies künstlerisch umsetzt. In drei Stunden programmieren die Teilnehmenden aus einfachen Formen eine Figur oder ein Muster, die sie mit dem Schneidplotter bzw. Vinyl-Cutter ausschneiden und dann auf einen Stoffbeutel aufbügeln können.
Dieses Format wurde mit 12 Mädchen im Alter von 12 bis 15 Jahren und drei Betreuer*innen erfolgreich durchgeführt. Materialien zur Durchführung des Workshops sind im EduFab - Koffer in der StoffFab - Ebene zu finden (siehe Anhang A).

2. Inhalt des Workshops: Programmierung mit Processing

Inhaltlich steht die Programmierung eines Musters oder einer Figur im Vordergrund. Der Schneidplotter wird nur kurz thematisiert, da er am Ende eher als Werkzeug benutzt wird.

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

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

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

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 des Workshops

3.1. Bereitstellen des Konstruktionsmaterials

Das für den Workshop benötigte Material ist im EduFab-Koffer bzw. in den dazugehörigen Listen aufgeführt. Eine Kopie der relevanten Seiten der Kofferbeschreibung befindet sich in Anhang A.

Auf einem Materialtisch können Folie und Stoffbeutel ausgelegt werden. Als Werkzeuge sollten Abbrechmesser und Bastelmesser bereit stehen. Zur Ideenfindung werden Moosgummiteile, Millimeter- bzw. weißes Papier, Stifte und Radiergummis benötigt. Die Processing-Handreichungen werden bereitgestellt. In unserem Arbeitsumfeld, in dem der Vinyl-Plotter von einem zentralen Rechner aus angesteuert wird, haben sich USB-Sticks (in den Händen der Durchführenden) zur Übertragung der Dateien bewährt.

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 sdxf-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.

In unserem Arbeitsumfeld, in dem der Vinyl-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. Was ist ein Schneidplotter bzw. Vinyl-Cutter und wie funktioniert er?

In diesem Teil wird kurz erklärt, was der Vinyl-Cutter macht, nämlich dass er mit einem Messer Linien ausschneidet. Im Gegensatz zu einem normalen Drucker, bei dem von oben nach unten Zeilen von links nach rechts gedruckt werden, werden hier Linien unabhängig von einer Richtung geschnitten. Aus diesem Grund braucht es eine spezielle Grafikdatei (statt einer Bilddatei aus Pixeln) - eine sogenannte Vektorgrafik. Diese lässt sich auf verschiedene Weise erzeugen, heute soll sie programmiert werden.

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(600, 600)* bewährt, siehe Zeile 1 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).

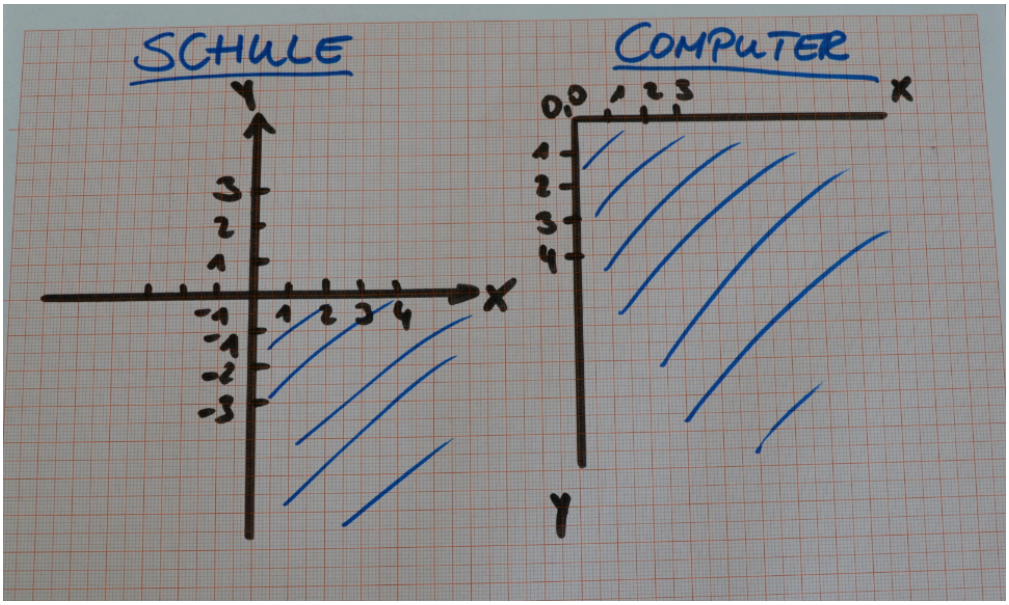


Abbildung 1: Vergleich der Aufteilung eines Koordinatensystems und des Processing-fensters

Der Hintergrund dieses Fensters wird mit dem Befehl *background(255)* (siehe Zeile 2 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 4 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 5 in Quellcode 1). Dieser wird ebenso zunächst am Whiteboard notiert und veranschaulicht und im Anschluss am Rechner ausgeführt.

```
1    size (600, 600);
2    background (255);
3
4    line (30 ,20 ,85 ,75);
5    line (85 ,75 , 110 ,50);
```

Quellcode 1: Programmierung in der Einführung

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.

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 plotten und aufzubügeln.

4.7. Erzeugen einer Grafikdatei

Bevor die entstandene Grafik zum Vinyl-Plotter geschickt werden kann, muss die Ausgabe in eine dxf-Datei erfolgen[2] (siehe Quellcode 2). Diese Aufgabe haben bei uns die Tutor*innen übernommen, da die einzelnen Schritte für Programmieranfänger*innen schwer nachzuvollziehen und in diesem Szenario nicht relevant sind. Die Teilnehmenden übergeben hier ihren fertigen Quellcode per USB-Stick an den Tutor oder die Tutorin.
Zur Erzeugung muss zunächst in Zeile 1 ein die entsprechende Bibliothek eingebunden werden. Die Nutzung der Bibliothek geschieht in der *setup()*-Funktion, die in Zeile 3 geöffnet und in Zeile 18 geschlossen wird. Der *size()*-Befehl erhält einen dritten Parameter (P3D), mit dem der hier notwendige Renderer benutzt wird. In Zeile 7 erfolgt der Befehl der die Ausgabe startet und sie in die dxf-Datei leitet. In Zeile 17 folgt der Befehl zum Beenden der Ausgabe.

```
1   import processing.dxf.*;
2
3   void setup() {
4
5   size(600,600, P3D);
6   background(255);
7   beginRaw(DXF,"pizza.dxf");
8
9   triangle(24,65,208,48,142,290);
10  triangle(50,95,190,80,140,270);
11  ellipse(106,148,30,30);
12  ellipse(150,110,30,30);
13  ellipse(82,110,30,30);
14  ellipse(135,215,30,30);
15  ellipse(145,168,30,30);
16
17  endRaw();
18  }
```

Quellcode 2: Erzeugung einer Grafikdatei mit Processing

[2] Bitte die Processingversion 2.x.x verwenden

4.8. Fertigstellen der Stoffbeutel

Die fertige dxf-Datei kann dann in die Silhouette Software geladen werden. Diesen Teil übernimmt ein*e Tutor*in gemeinsam mit den Teilnehmenden. Dort wird gemeinsam die genaue Größe festgelegt. Der Schneidevorgang wird gestartet und die Teilnehmenden erhalten ihre Form, die sie dann noch von den Teilen freilegen müssen, die nicht aufgebügelt werden sollen (entgittern).

Gemeinsam mit einem oder einer weiteren Tutor*in wird der Bügelvorgang vorbereitet und durchgeführt. Nach dem Abkühlen können die Teilnehmenden dann die letzte Folienschicht vom Beutel lösen und ihr Werk ist fertig.

4.9. Präsentation

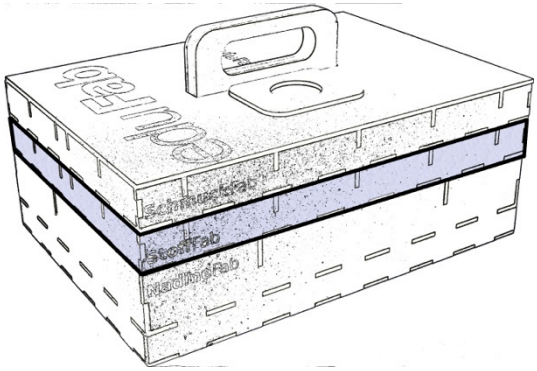
Am Ende präsentieren die Teilnehmenden ihre Stoffbeutel gegenseitig. Sie erhalten dabei Anerkennung für ihre Arbeit seitens der anderen Teilnehmenden sowie der Tutor*innen.

4.10. Abschluss: Bezug zum Alltag

Am Ende kann kurz darauf eingegangen werden, an welchen Stellen im Alltag Programmierung zum Einsatz kommt. Häufig werden interaktive Dinge programmiert, z.B. Apps, Ampelschaltungen oder Roboter.
Programmierung kann aber auch künstlerisch eingesetzt werden – so wie in diesem Beispielworkshop. Bei Grafiken ergibt dies insbesondere dann Sinn, wenn sich wiederholende Muster erstellt werden sollen (sonst können sie auch mit einem Zeichenprogramm erstellt werden). Im Workshop gibt es einen kleinen Einblick in Programmierung, der „getragen“ werden kann.

Anhang A: Die „StoffFab“ – Ebene des EduFab - Koffers

Die „StoffFab“ - Ebene



Die „StoffFab“ - Ebene enthält Materialien, mit denen mit dem **Vinyl-Cutter Textildrucke** (z.B. für Jutebeutel oder T-Shirts) gefertigt werden können.

Die „StoffFab“ – Ebene dient der **Durchführung des Kurzworkshops „Stoffdrucke programmieren“**. In diesem Workshop werden mit der Programmiersprache Processing Formen programmiert, die mit dem Vinyl-Plotter aus Folie ausgeschnitten und auf Stoffbeutel gebügelt werden.

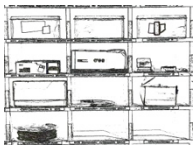
Der Workshop „Stoffdrucke programmieren“ ähnelt inhaltlich sehr stark den „Programmierte(n) Schmuckstücke(n)“. Die Idee dahinter ist, dass dieser Workshop die „Unterwegs-Version“ bzw. die kostengünstigere Variante darstellt, da kein großer Laser-Cutter, sondern ein eher handlicher und preiswerter Vinyl-Cutter gebraucht wird.

Die im Koffer vorhandenen Materialien der „StoffFab“ – Ebene umfassen:

- Einfache Formen (z.B. Moosgummiteile) zur Ideenfindung,
- ein Abbrechmesser und Ersatzklingen zum Schneiden der Folien,
- verschiedene Bastelmesser und Ersatzklingen zum Entfernen der überflüssigen Folie vor dem Bügeln,
- sowie ein Ersatzmesser für den Vinyl-Cutter, als Backup für die Arbeit mit dem Vinyl-Cutter.

Aus der <name>Fab – Ebene sind für den Stoffdrucke – Workshop zudem nützlich:

- Der USB-Stick zum Transfer der Schnittdateien,
- die Schere zum Schneiden von Folien,
- der Bleistift mit Anspitzer, Radierer und (Millimeter-) Papier zum Anfertigen von Skizzen.



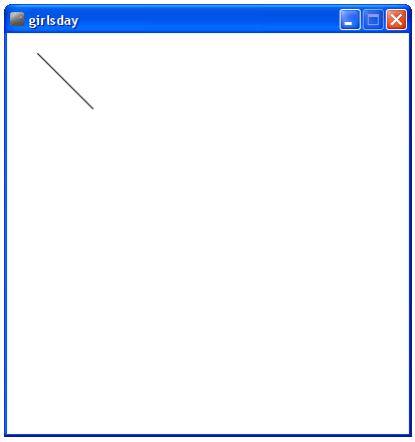
Im FabLab werden für die Anfertigung der Stoffbeutel nach dem beschriebenen Szenario weiterhin benötigt:

- Computer für die Programmierung und die Ansteuerung des Vinyl-Cutters,
- die freie Programmiersprache Processing zum Designen der Stoffdrucke,
- ein Vinyl-Cutter zum Ausschneiden der programmierten Formen,
- ein Programm zur Steuerung des Vinyl-Cutters, z.B. Silhouette Studio,
- Material, aus dem die Drucke gecuttet werden, z.B. Flexfolie oder Flockfolie,
- Stoffbeutel möglichst verschiedener Ausführungen und Farben, sowie
- eine Bügelpresse oder -eisen zum Aufbügeln der Folie auf dem Stoff.

Digital stehen für die Schmuckanfertigung folgende Dateien zur Verfügung:

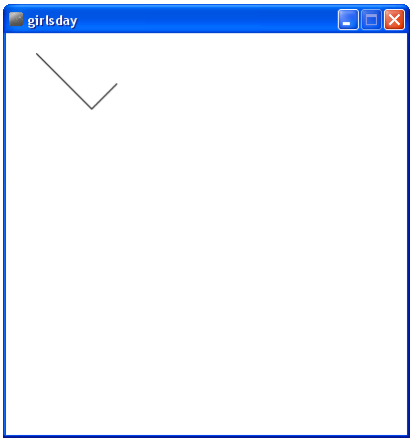
- das Konzept zur Durchführung des Kurzworkshops „Stoffdrucke programmieren“ *edufab_workshop_stoffdrucke.pdf*,
- ein Link zur Programmierumgebung Processing *download_processing.URL*, sowie
- ein Handout zur Programmierung mit Processing zum Ausdrucken und Verteilen *edufab_handout_stoffdrucke.pdf*.

1. Linien, Dreiecke und Vierecke



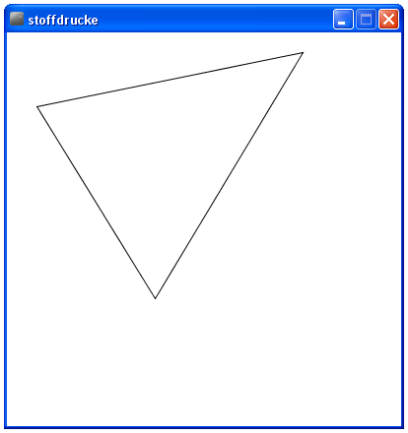

```
line(30, 20, 85, 75);
```

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



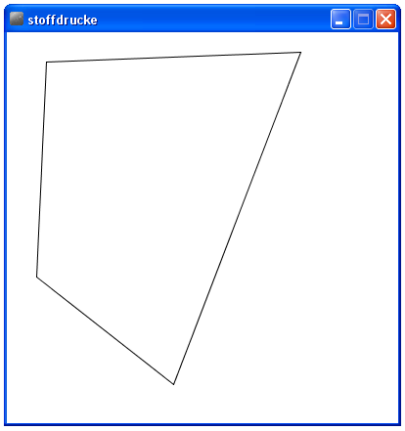

```
line(30, 20, 85, 75);
line(85, 75, 110, 50);
```

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);
```

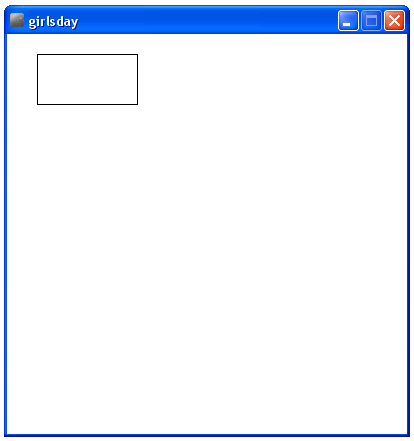
zeichnet ein geschlossenes **Dreieck**
(engl. *triangle*) von
Punkt (30, 75) = erster Eckpunkt
über Punkt (300, 20) = zweiter Eckpunkt
über Punkt (150, 270) = dritter Eckpunkt
zurück zum ersten Eckpunkt




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

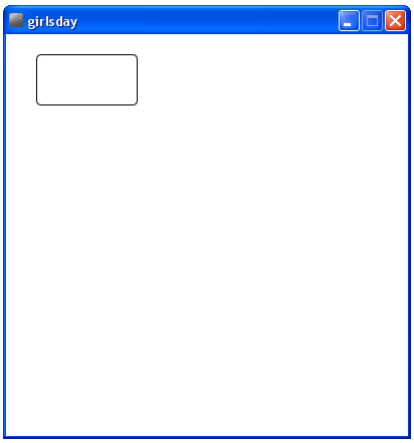
zeichnet ein geschlossenes **Viereck**
(engl. *quadrilateral*) von
Punkt (40, 30) = erster Eckpunkt
über Punkt (300,20) = zweiter Eckpunkt
über Punkt (170, 360) = dritter Eckpunkt
über Punkt (30, 250) = vierter Eckpunkt
zurück zum ersten Eckpunkt

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)

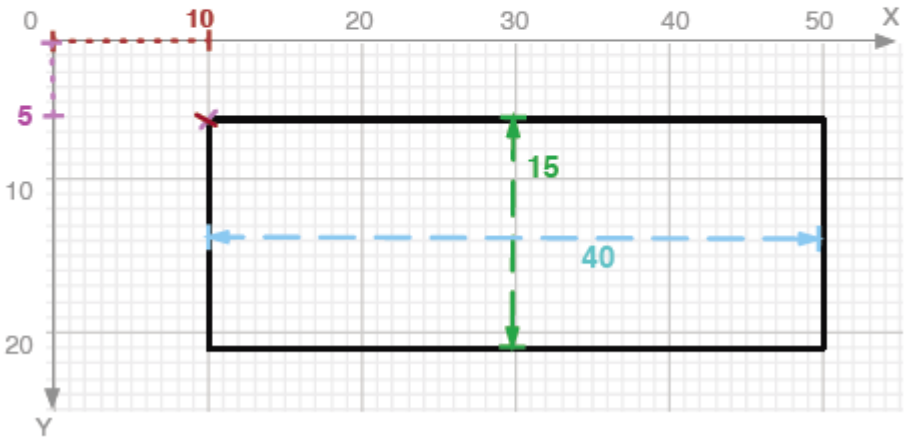



```
rect(30, 20, 100, 50,5);
```

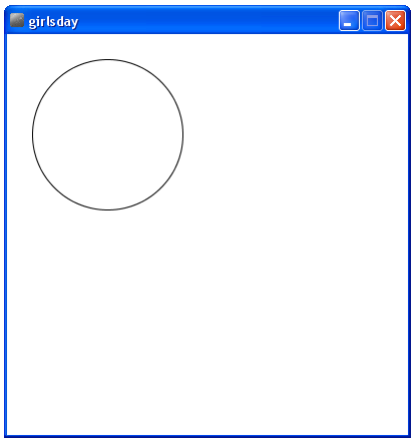
zeichnet ein Rechteck mit „runden“ Ecken

rect(10, 5, 40, 15);

rect(x, y, b, h);

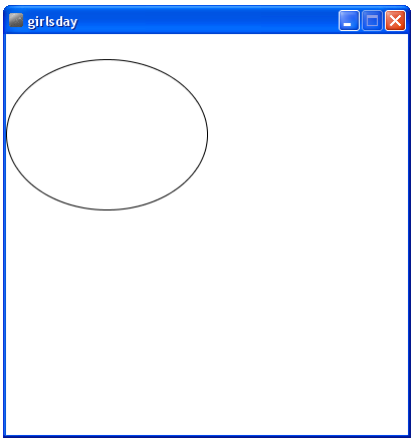


3. Kreise, Ellipsen und Bogen




```
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);