Weihnachtsbaumgenerator

Basics

Baum ....

Ständer ....

Baumschmuck

Beleuchtung
- Lichter
- Kerzen

Anhänger
- Kugel <stepper> <select>
- Glocke " "
- Stern " "
- Engel " "
- Herz " "
- Lametta " "
- Perlenkette " "

Natur
- ☐ Orangenschale
- ☐ Zimtsterne
- ☐ Zuckerstangen
- ☐ Zapfen

Spitze
- keine
- Engel color
- Stern color

<h1>
<form>
<fieldset>
<legend> </>
</>
</>

Lieferdaten

Versand
- DHL
- Hermes
- UPS

Persönliche Daten

Anrede [ ] <select>

Name

Vorname

Straße NR

PLZ Ort

E-Mail

Tel

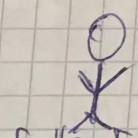
Nachricht <textarea>

Check Reset

<button type="submit">

<button type="reset">

Anwendungsdiagramm

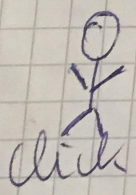


fülle Formular aus

Form-Element ändern Status

↳ wenn alle Form-Elemente für ein Item ausgefüllt sind

Einzel- & Gesamtpreis wird in Warenkorb angezeigt

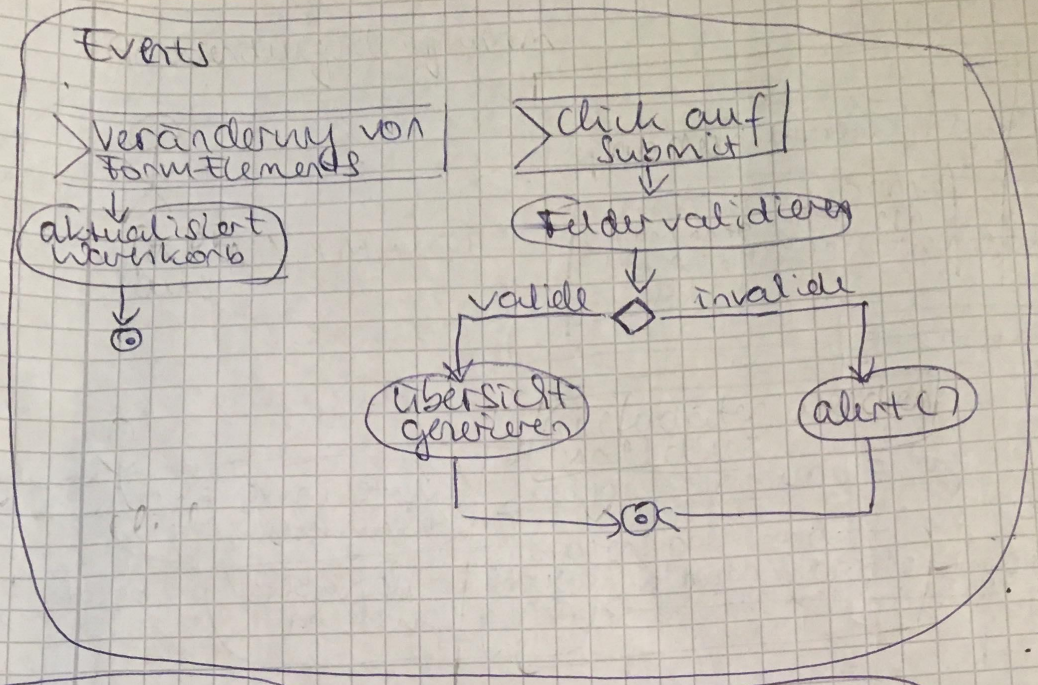


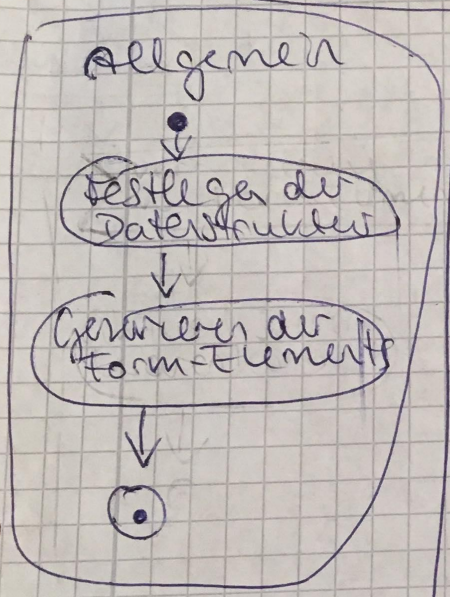
click Submit

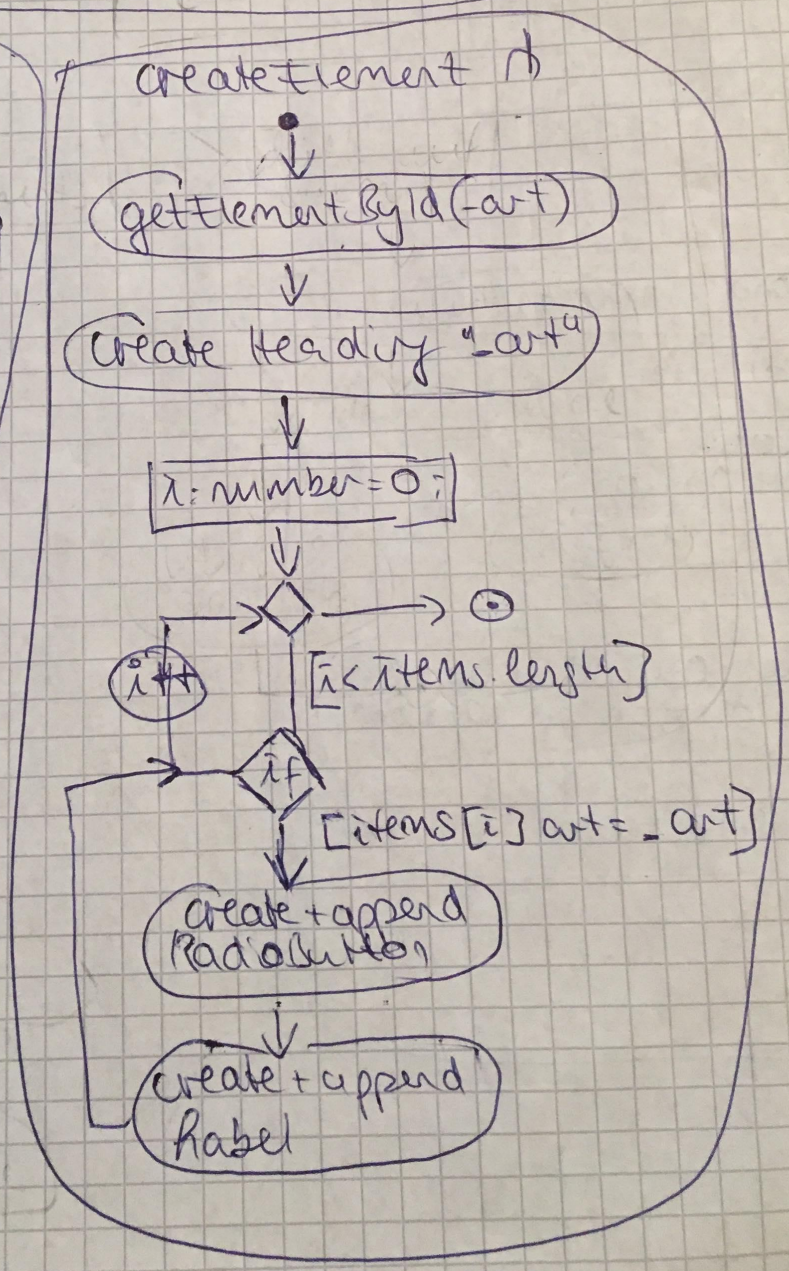
Liefdaten werden validiert

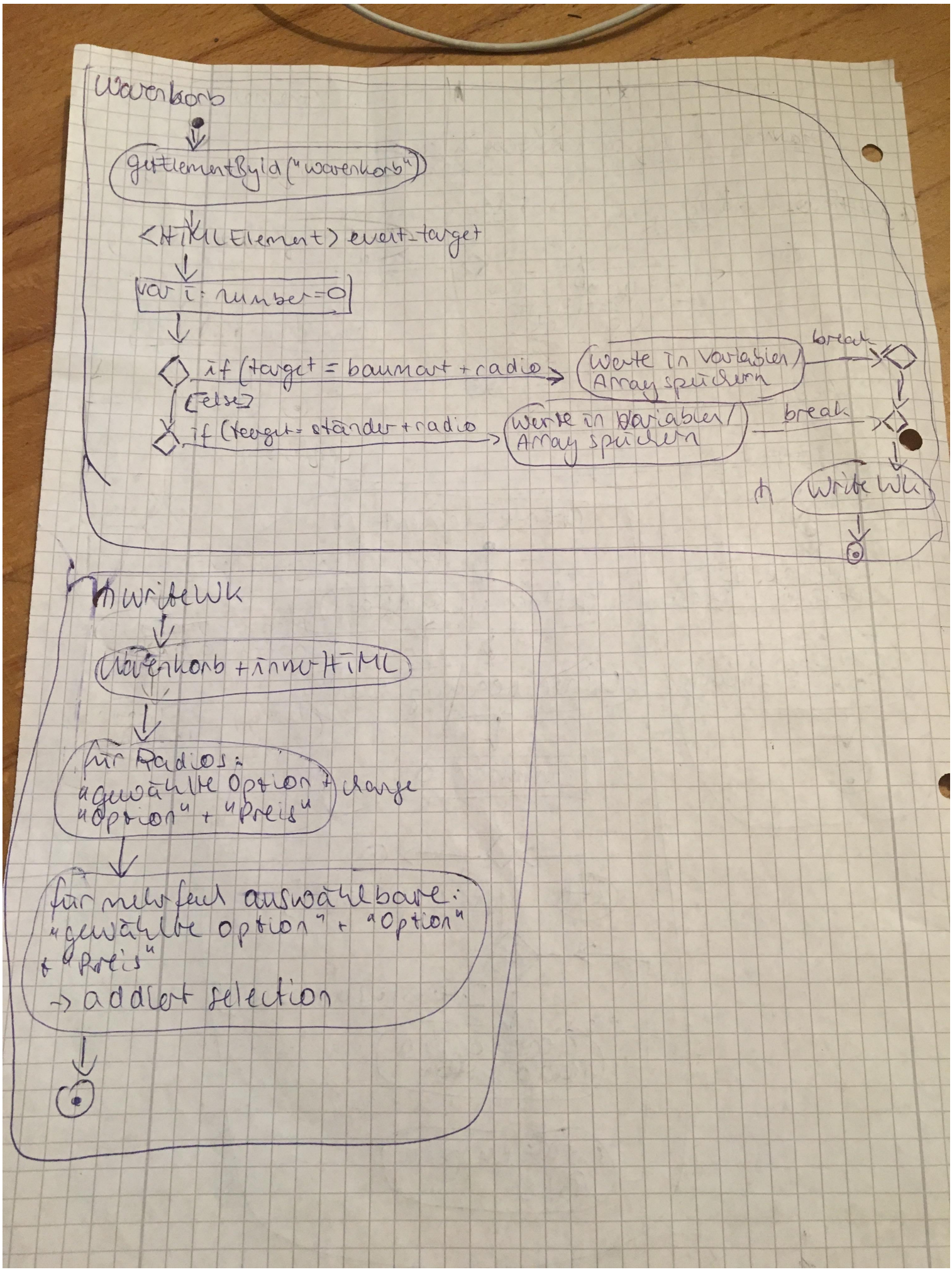
↳ Warenkorb-Inhalt + Liefdaten => Bestellübersicht

kann Bestellungen dann übersicht überprüfen


Events
Veränderung von Form-Elements
aktualisiert Warenkorb
Click auf Submit
Felder validieren
valide
invalide
Übersicht generieren
alert()


Allgemein
festlegen der Datenstruktur
Generieren der Form-Elements

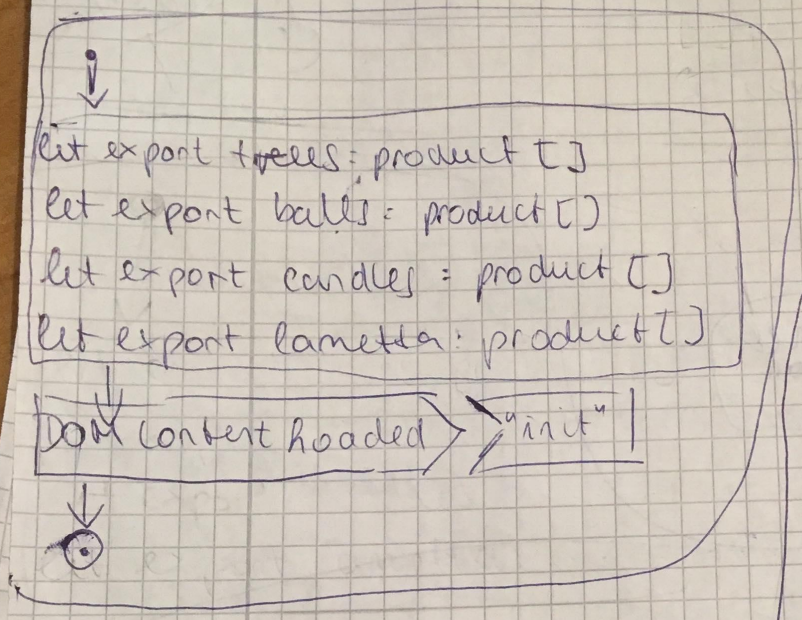

createElement()
getElementById(_art)
Create Heading "_art"
i: number = 0;
i++
[i < items.length]
if
[items[i].art = _art]
create + append Radiobutton
create + append Label

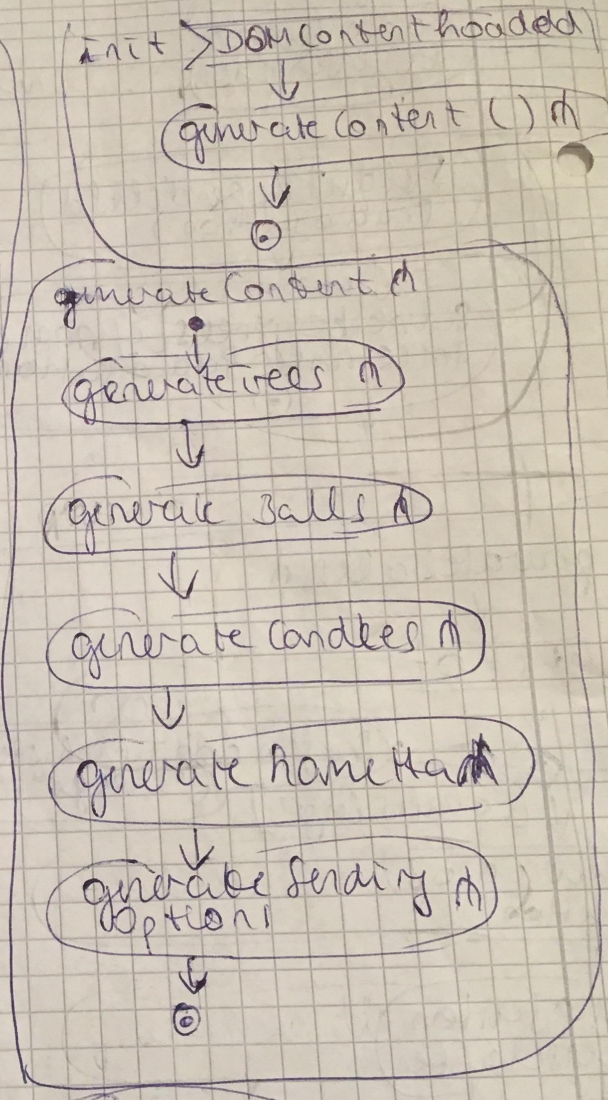

Warenkorb
getElementById("warenkorb")
<HTMLElement> event.target
var i: number=0
if (target = baumart + radio
Werte in Variablen/ Array speichern
break
[else]
if (target = ständer + radio
Werte in Variablen/ Array speichern
break
Write WK
writeWK
Warenkorb + innerHTML
für Radios:
"gewählte Option" + change "Option" + "Preis"
für mehrfach auswählbare:
"gewählte Option" + "Option" + "Preis"
-> addert selection

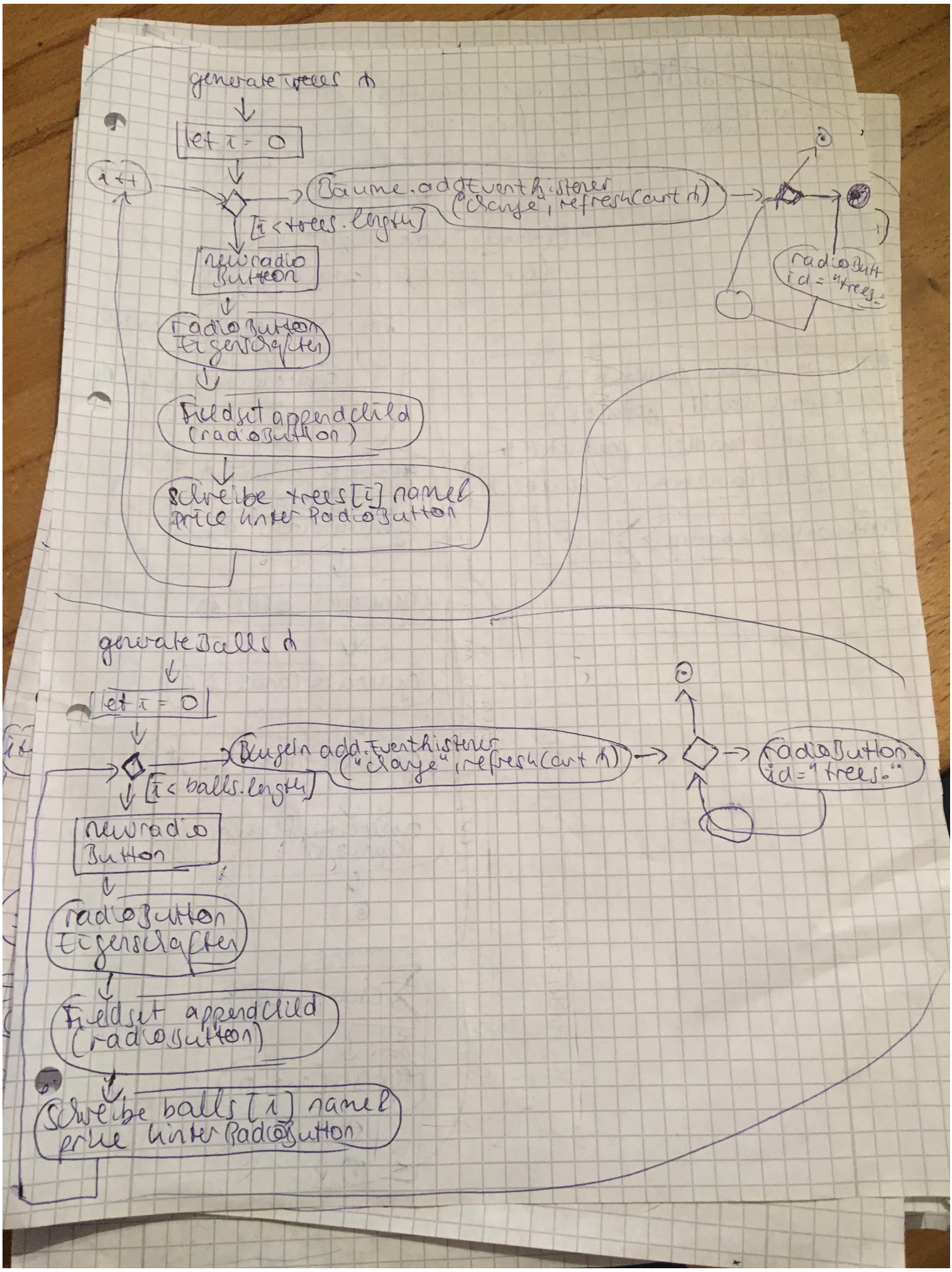
Datenstrukturen
namespace WBL2
interface product{
name: string;
price: number;
}

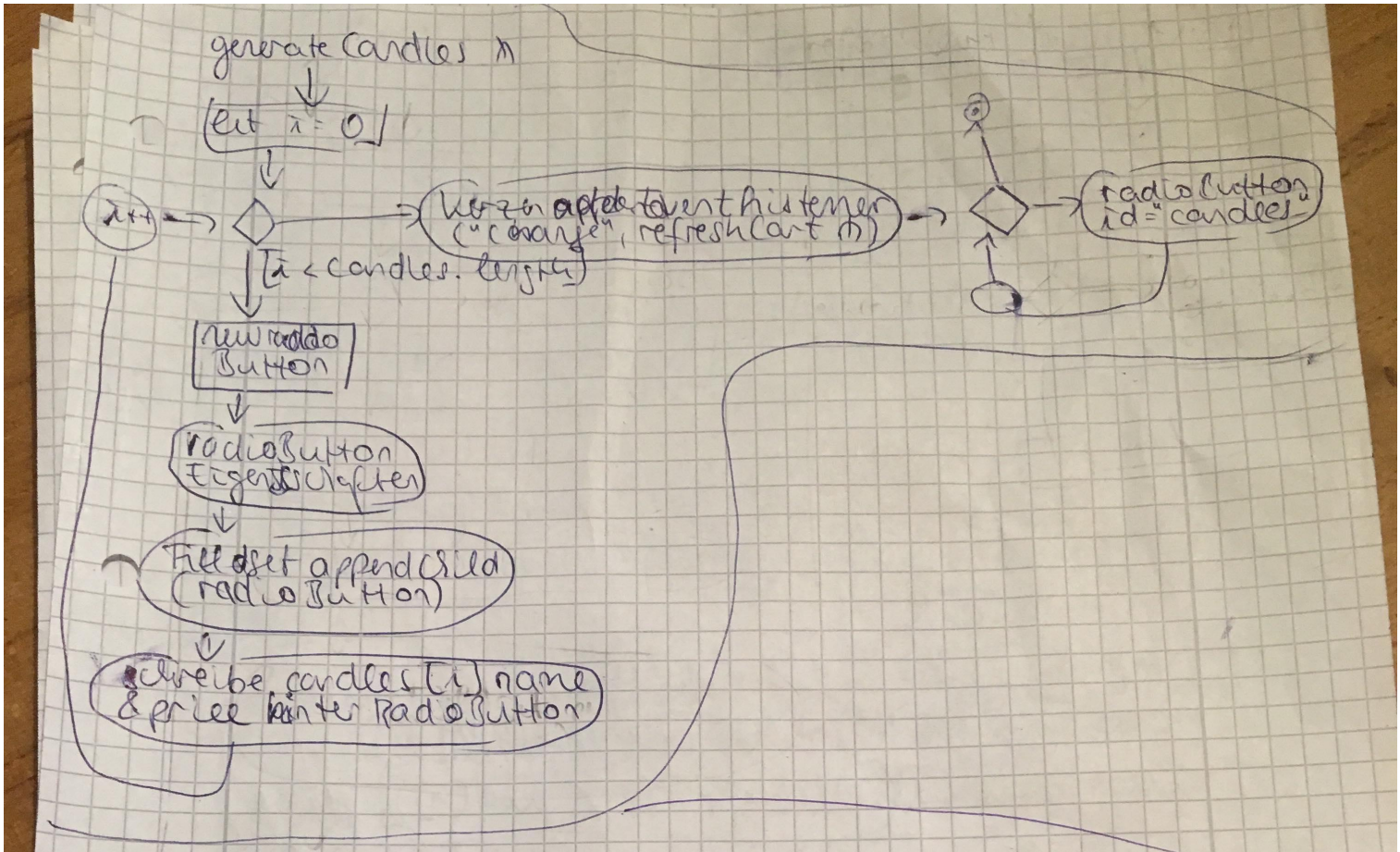
let trees: product[ ]=
[{name: "Tanne", price: 39,98}
{name: "Fichte", price: 49,98}

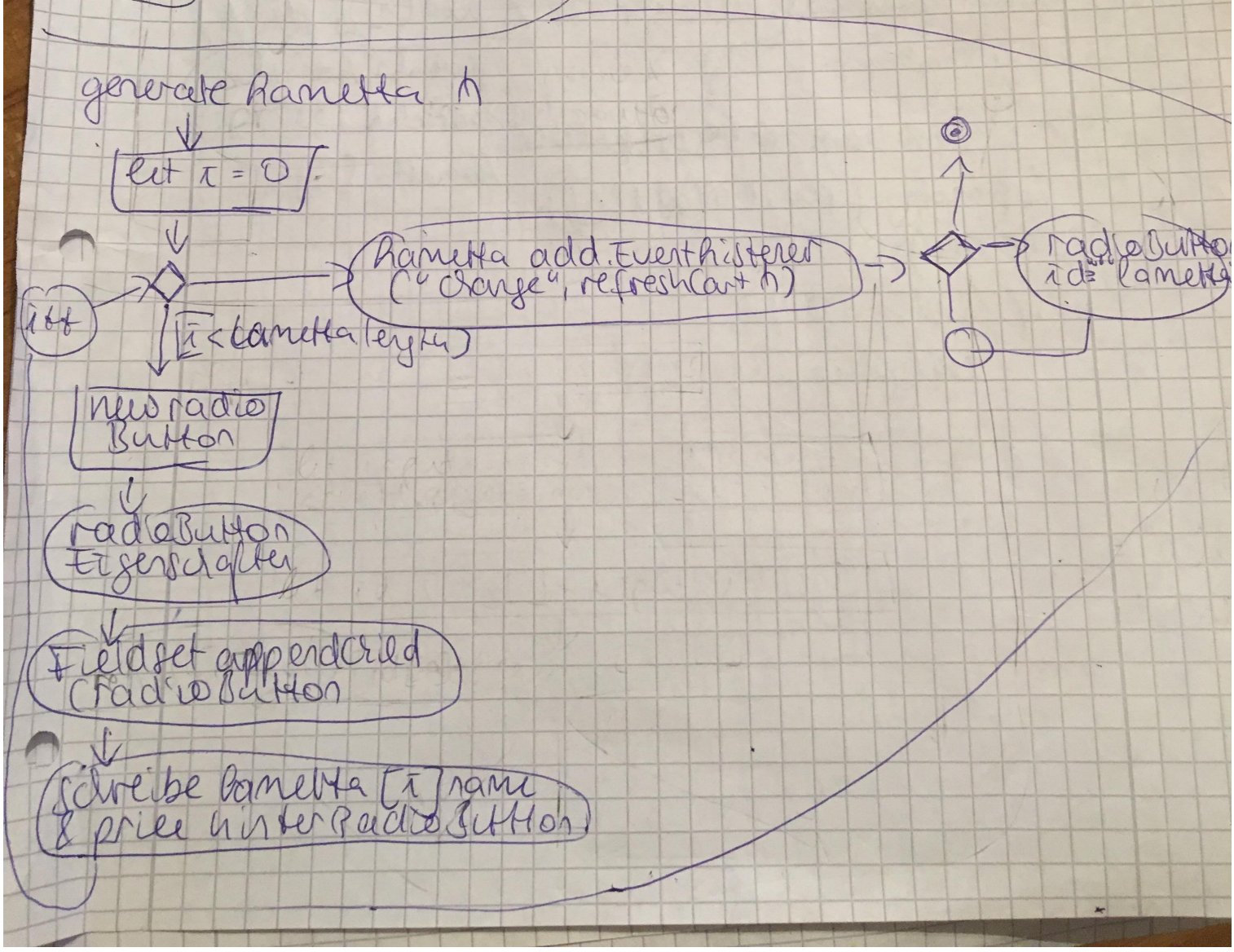
let balls: product[ ]=
[{name: "Rote Kugel", price: 3,00}
let candles: product[ ]=[ ];

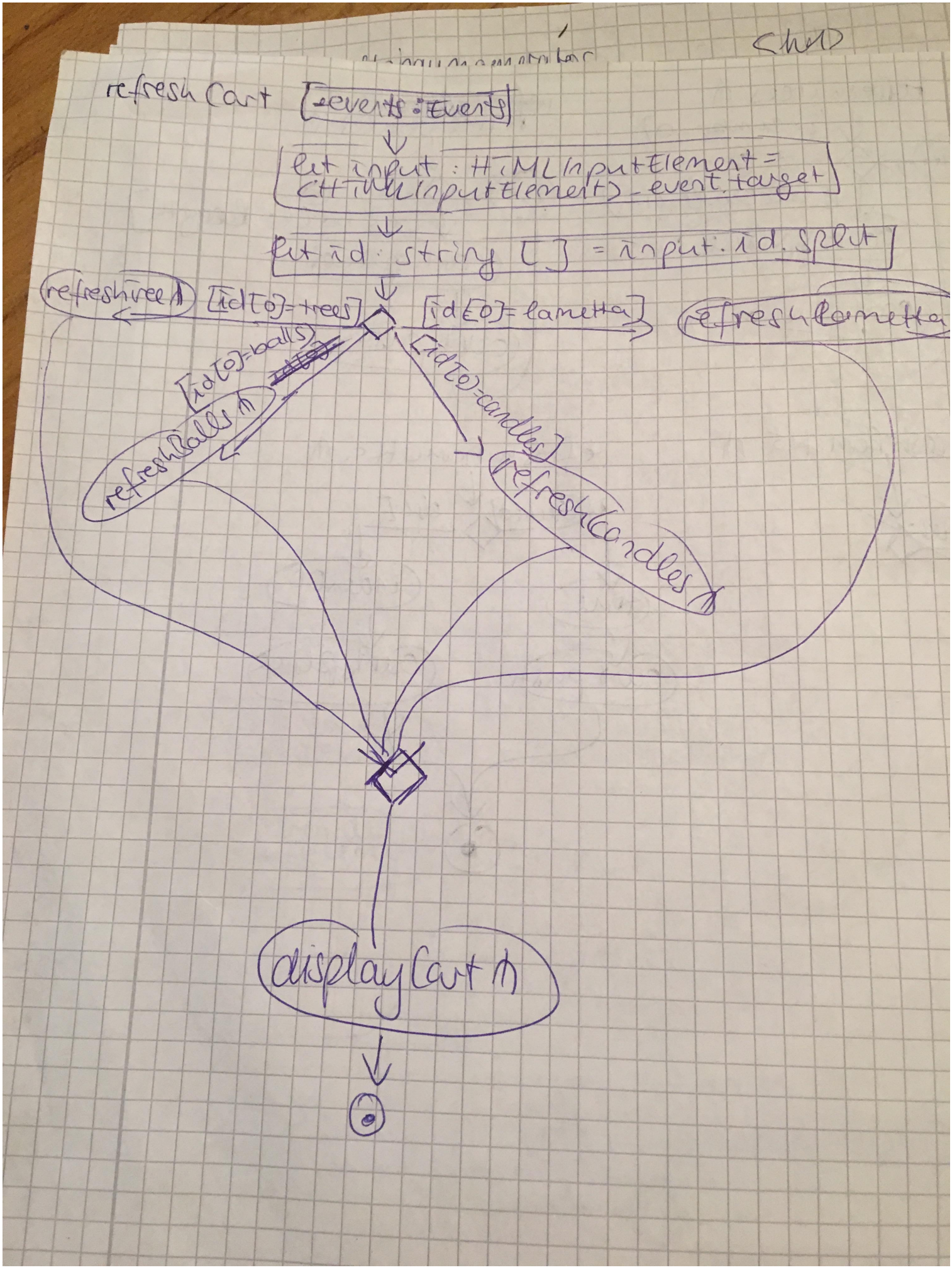


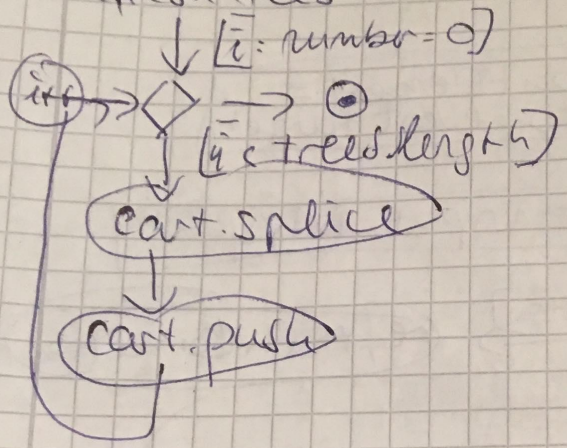


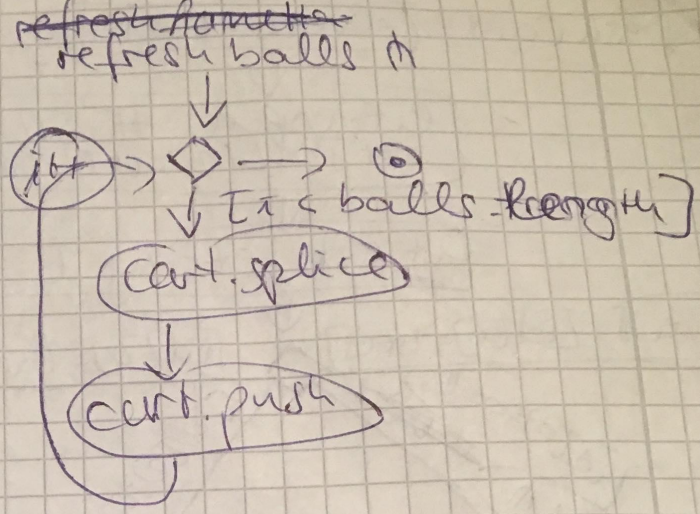

generateTrees()
let i = 0
i++
[i < trees.length]
Bäume.addEventListener ("change", refresh(cart()))
radioButt id = "trees"
newradioButton
radioButton Eigenschaften
Fieldset appendChild (radioButton)
schreibe trees[i].name/preis hinter RadioButton
generateBalls()
let i = 0
[i < balls.length]
Bälle.addEventListener ("change", refresh(cart()))
radioButton id = "trees"
newradioButton
radioButton Eigenschaften
Fieldset appendChild (radioButton)
schreibe balls[i].name/preis hinter RadioButton

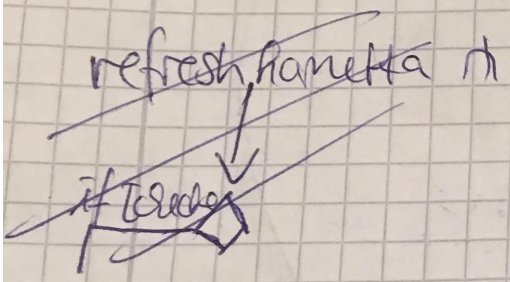

generate Candles n
let i = 0
i++
Kerzen add.EventListener ("change", refreshCart n)
radioButton id="candles"
[i < candles.length]
new radio Button
radioButton Eigenschaften
Fieldset appendChild (radioButton)
schreibe candles[i] name & price hinter RadioButton

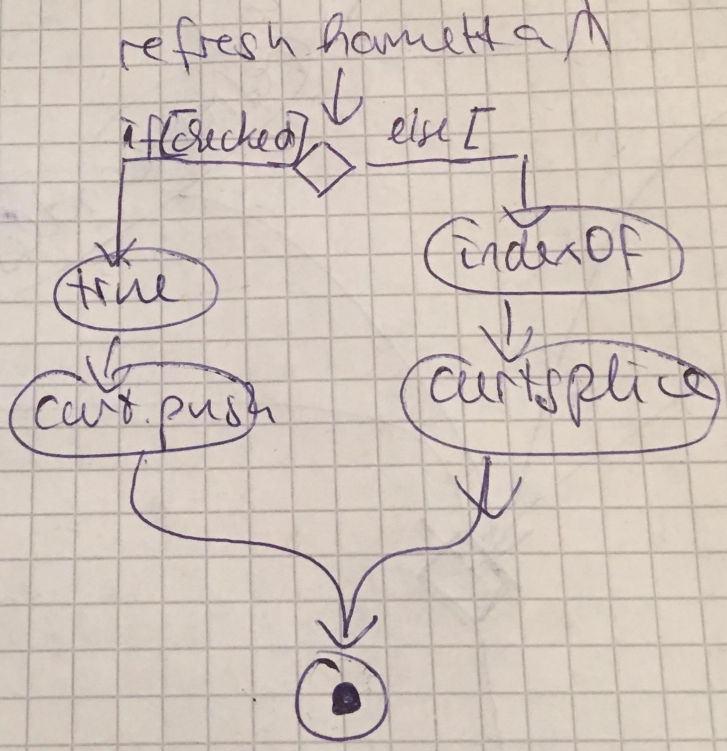

generate Kametta n
let i = 0
i++
Kametta add.EventListener ("change", refreshCart n)
radioButton id="kametta
[i < kametta length]
new radio Button
radioButton Eigenschaften
Fieldset appendChild (radioButton
schreibe kametta[i] name & price hinter RadioButton


refreshCart [events : Events]
let input : HTMLInputElement = (HTMLInputElement) event.target
let id : string [] = input.id.split
refreshTree() [id[0]=trees]
[id[0]=lamettia]
refreshLamettia
[id[0]=balls]
refreshBalls()
[id[0]=candles]
refreshCandles()
displayCart()


refreshTrees()
[i: number = 0]
i++
[i < treesLength]
cart.splice
cart.push


~~refreshHametta~~
refreshBalls()
i++
[i < balls.length]
cart.splice
cart.push


~~refreshHametta()~~
~~if [checked]~~


refreshHametta()
if [checked]
else [
true
cart.push
indexOf
cart.splice