



Yoshi Malaise
ymalaise@vub.be
 0000-0002-3228-6790



FOSDEM 2023

Breaking the Code to Inclusion

Designing Micro Materials Based on PRIMM Principles for Accessible Programming Education



VRIJE
UNIVERSITEIT
BRUSSEL

 WEB & INFORMATION
SYSTEMS ENGINEERING

Introduction





HACK
YOUR
FUTURE

Background

demo – Main.java

demo > src > main > java > com > example > demo > Main > main

Project Commit Notifications Maven

demo ~/Downloads/ymalaise-thesis-2020-master/code/1 pom.xml (demo) × hello-view.fxml × HelloController.java × HelloApplication.java × Main.java ×

1 package com.example.demo;
2
3 public class Main {
4 public static void main(String[] args) {
5 System.out.println("Hello world!");
6 }
7 }
8

src
main
java
com.example.demo
HelloApplication
HelloController
Main
module-info.java
resources
target
.gitignore
demo.iml
mvnw
mvnw.cmd
pom.xml
External Libraries
Scratches and Consoles

Build: Sync × Build Output ×

demo: build failed At 22/11/2022, 10:15 with 1 error 1 sec, 219 ms

Main.java src/main/java/com/example/demo 1 error
';' expected :5

/Users/yoshimalaise/Downloads/ymalaise-thesis-2020-master/code/rsl-backend/demo/src/main/java/com/example/demo/Main.java: ';' expected

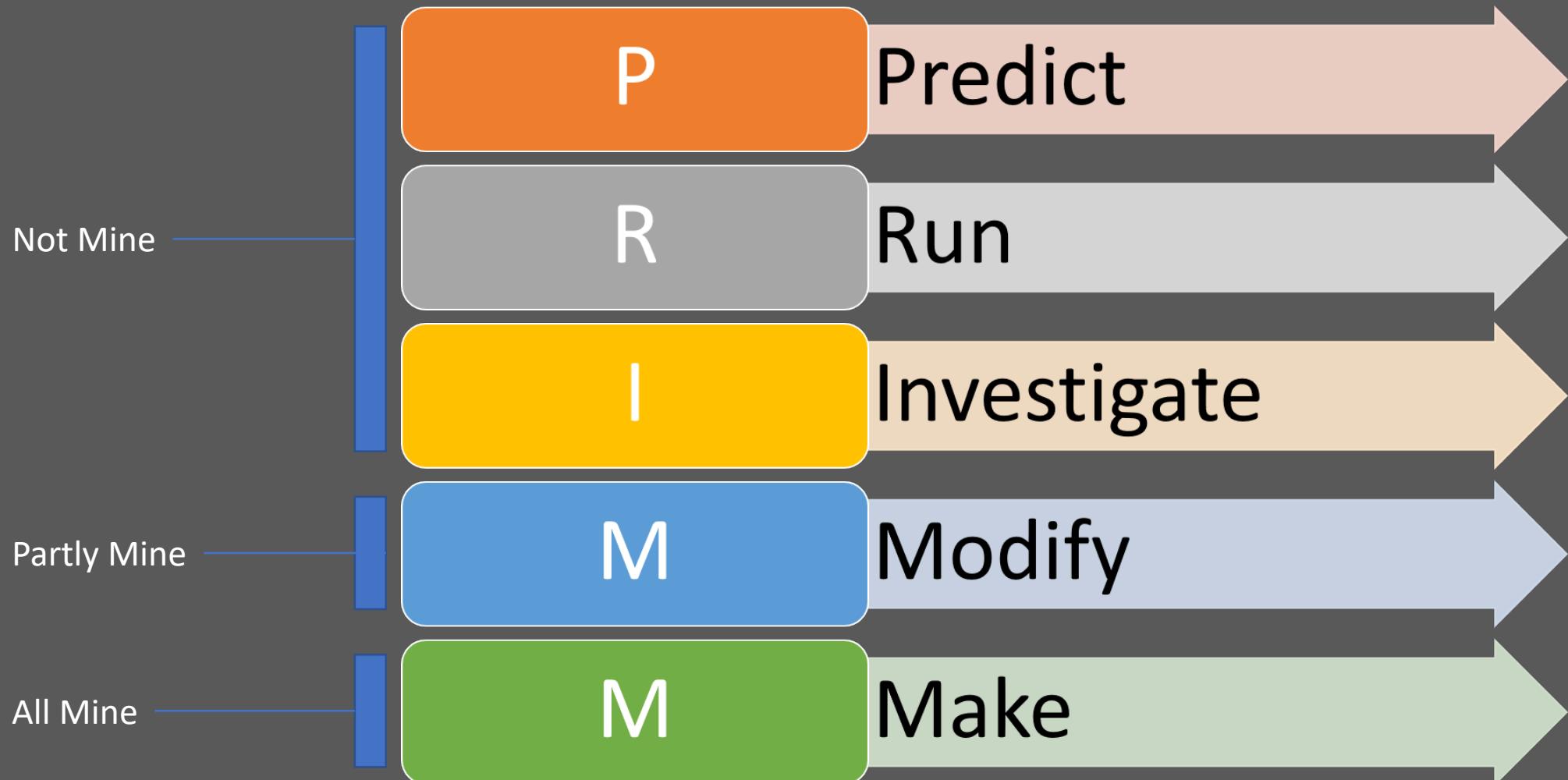
Structure Bookmarks

Git Run TODO Problems Terminal Services Build Dependencies

:' expected 5:43 LF UTF-8 4 spaces main

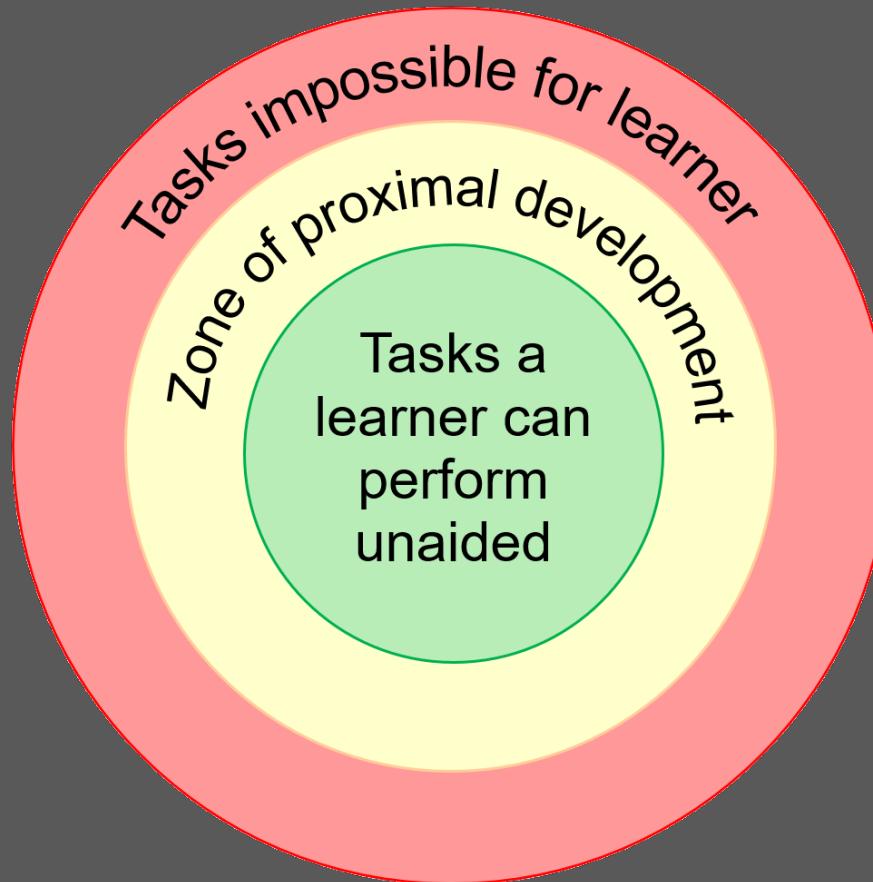


PRIMM



- Sentance et al., 2019

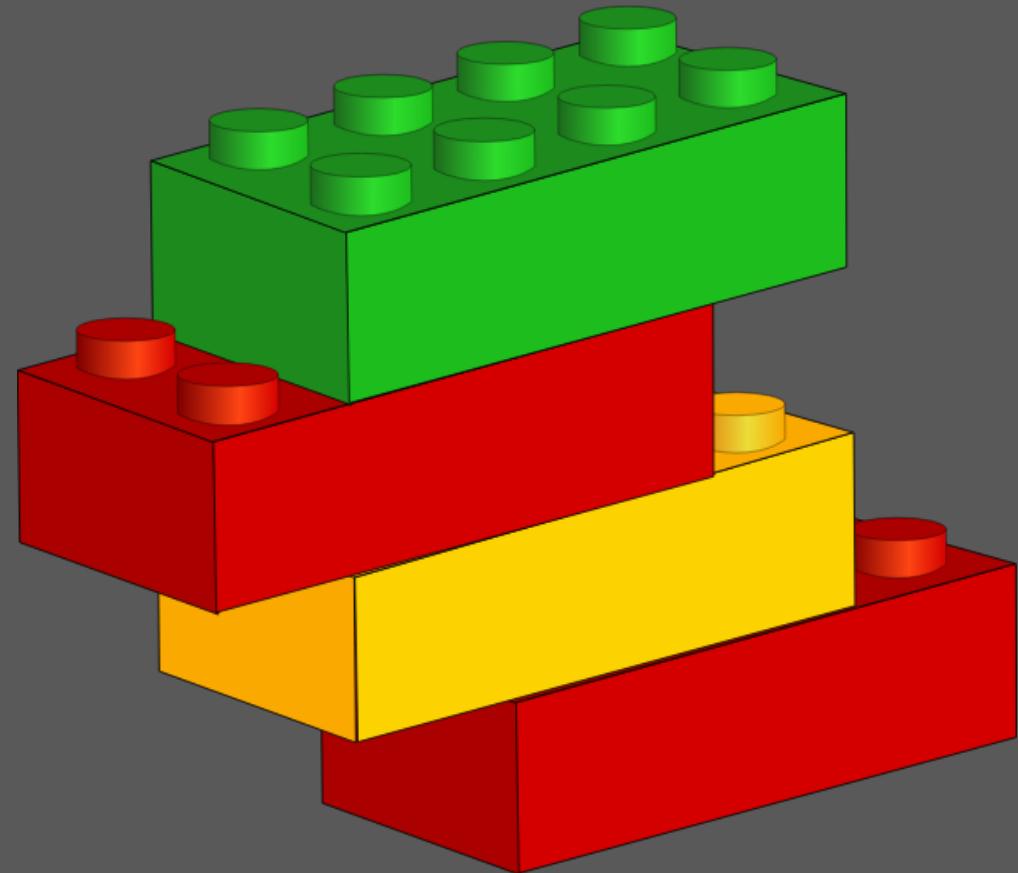
Zone of Proximal Development



- Lev Vygotsky., 1978

Micromaterials

- Open Education Resources
- Should be easy to integrate in existing curricula
- Should provide automated feedback
- Ideally content should be generated automatically



- Adam Leskis., 2008

Examples

HTML StuddyBuddy

html-studybuddy.netlify.app

← Level Select

Design Selection

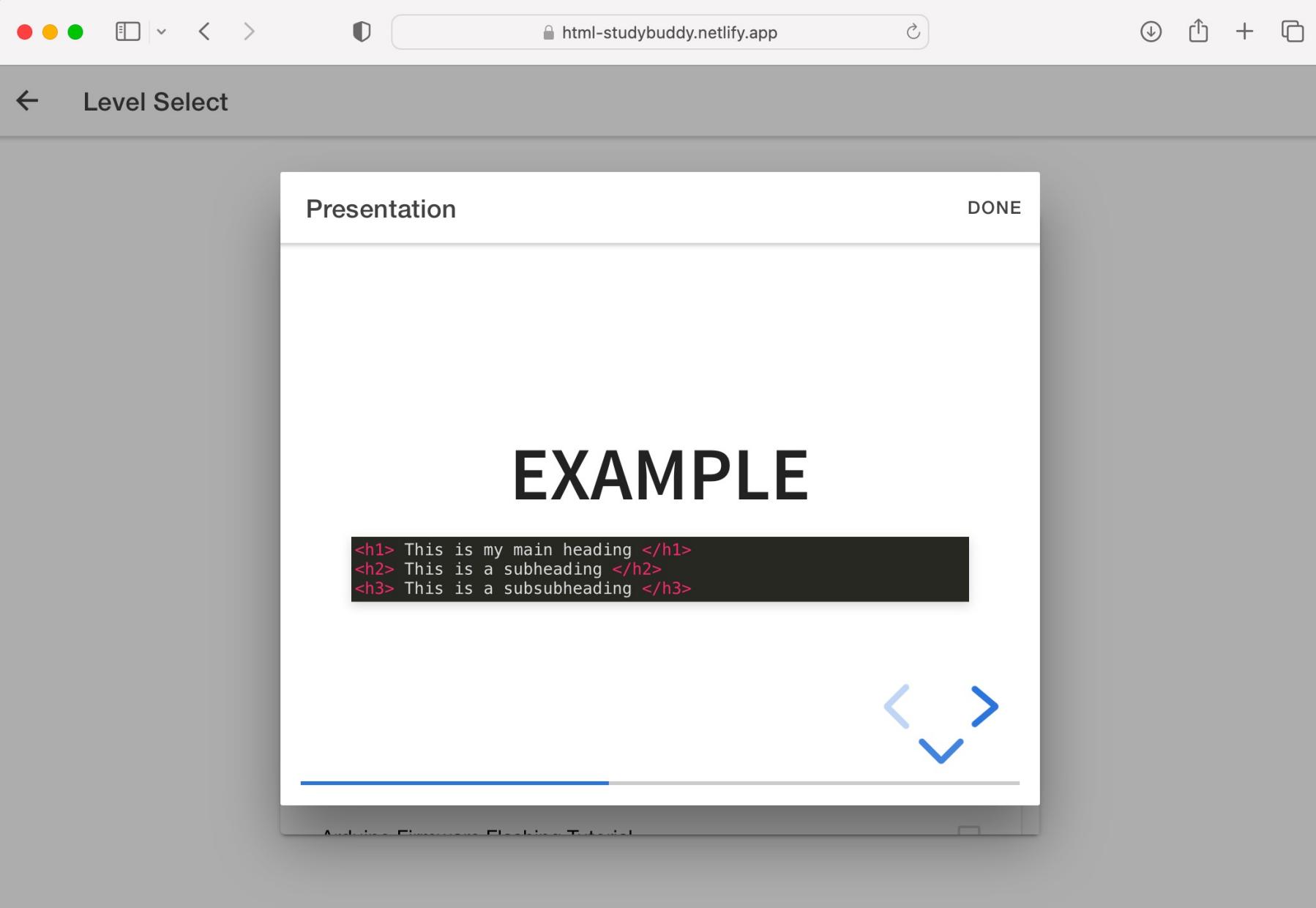
Select the design you want to implement.

Titles and Paragraphs

- Welcome Party 
- Employee Manual 
- About WISE 

Lists

- Shopping List 
- Muffin Recipe 



The screenshot shows a web browser window with the URL `html-studybuddy.netlify.app`. The title bar says "Welcome Party". On the left, there's a sidebar with two tabs: "headings" (grayed out) and "content" (selected, highlighted in blue). Below the tabs is a code editor showing the following HTML structure:

```
<!DOCTYPE html>
<html>
  <head>
  </head>
  <body>
    <h1>
      Welcome party
    </h1>
  </body>
</html>
```

The main content area has a light gray background and contains the heading "Welcome party" and some placeholder text.

This screenshot is similar to the first one, but it shows the "content" tab selected in the sidebar, which highlights the text elements in the code editor. The code editor now displays several paragraphs of text:

```
<p></p>
<html>
  <p>No actual design yet. But we have a few cool paragraphs here, try to replicate them.</p>
  <p>Morbi dictum placerat augue. Phasellus sit amet venenatis mi. Curabitur vitae nisl eu diam hendrerit porta. Mauris sagittis blandit eros a efficitur.</p>
  <p>Fusce vel magna placerat, mollis nunc eu, semper odio. Vestibulum ut velit a lacus consectetur fringilla auctor et ipsum. Cras aliquam id nibh sodales viverra. Etiam non placerat elit. Duis tempus eros quis augue scelerisque, placerat cursus arcu pharetra. Aliquam convallis, mauris id auctor tempus, odio eros molestie turpis, ac porttitor ante risus ac nisi. Donec convallis maximus purus ac dictum. Praesent convallis varius nisi, sed pretium ante sollicitudin quis. Aliquam molestie massa a mattis sagittis. Phasellus ac tortor justo. Nulla in libero fermentum, rhoncus justo et, dignissim ante. Suspendisse ultrices elementum enim, sed lacinia orci ullamcorper vitae. Nullam at pretium libero. Nulla sit amet elit vitae magna dignissim efficitur. Curabitur sit amet lorem a mauris fringilla convallis.</p>
  <p>No actual design yet. But we have a few cool paragraphs here, try to replicate them.</p>
  <h1>
    Welcome party
  </h1>
  <p>Morbi dictum placerat augue. Phasellus sit amet venenatis mi. Curabitur vitae nisl eu diam hendrerit porta. Mauris sagittis blandit eros a efficitur.</p>
  <p>Fusce vel magna placerat, mollis nunc eu, semper odio. Vestibulum ut velit a lacus consectetur fringilla auctor et ipsum. Cras aliquam id nibh sodales viverra. Etiam non placerat elit. Duis tempus eros quis augue scelerisque, placerat cursus arcu pharetra. Aliquam convallis, mauris id auctor tempus, odio eros molestie turpis, ac porttitor ante risus ac nisi. Donec convallis maximus purus ac dictum. Praesent convallis varius nisi, sed pretium ante sollicitudin quis. Aliquam molestie massa a mattis sagittis. Phasellus ac tortor justo. Nulla in libero fermentum, rhoncus justo et, dignissim ante. Suspendisse ultrices elementum enim, sed lacinia orci ullamcorper vitae. Nullam at pretium libero. Nulla sit amet elit vitae magna dignissim efficitur. Curabitur sit amet lorem a mauris fringilla convallis.</p>
</html>
```



← Welcome Party



headings

```
<!DOCTYPE html>
<html>
  <head>
  </head>
  <body>
```

content

```
</body>
</html>
```

Hints

DONE

This is the very first design that should get you used to using headings and paragraphs

Here are some element you will probably need:

`<p>` element

The `<p>` HTML element represents a paragraph. Paragraphs are usually represented in visual media as blocks of text separated from adjacent blocks by blank lines and/or first-line indentation, but HTML paragraphs can be any structural grouping of related content, such as images or form fields.

[Read more on the documentation page](#)

`<h1>` element

The `<h1>` to `<h6>` HTML elements represent six levels of section headings. `<h1>` is the highest section level and `<h6>` is the lowest.

[Read more on the documentation page](#)

e party

gn yet. But we have a few
ns here, try to replicate

placerat augue. Phasellus sit
s mi. Curabitur vitae nisl eu
t porta. Mauris sagittis
efficitur.

Ina placerat, mollis nunc eu,
Vestibulum ut velit a lacus
ingilla auctor et ipsum. Cras
h sodales viverra. Etiam non
uis tempus eros quis augue
lacerat cursus arcu pharetra.
allis, mauris id auctor
eros molestie turpis, ac
isus ac nisi. Donec convallis
s ac dictum. Praesent
s nisi, sed pretium ante
s. Aliquam molestie massa a
. Phasellus ac tortor justo.
fermentum, rhoncus justo et,
. Suspendisse ultrices
im, sed lacinia orci
tae. Nullam at pretium libero.
elit vitae magna dignissim
itur sit amet lorem a mauris
llis.

SQL StuddyBuddy



sql-studybuddy.netlify.app



SQL StudyBuddy

WiSE WEB & INFORMATION
SYSTEMS ENGINEERING

SQL StudyBuddy

An easy way to practise SQL

DATA QUERY LANGUAGE

DATA MANIPULATION LANGUAGE

DATA DEFINITION LANGUAGE

Code Editor

```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40
```

Wise Wishlist DB

DONE

items
id: number (PK)
name: varchar
price: float
member: varchar

The WISE Wishlist database has only table, items.

In this items table the data is stored for all items that WISE lab members might want to buy in the near future for their research purposes. Every entry in the table contains an id for the item, the name of the item, the price of the item and the first name of the WISE member that wants to buy the item.

32GB¹ to the wishlist, the model costs 2599\$.

at WISE lab members might want to buy
ry entry in the table contains an id for
n and the first name of the WISE



Code Editor



```
1
2 select * from items
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
```

Question

What is the price of the oscilloscope (use . for decimals)?

Answer:

SUBMIT

Query Result

Showing max 10 results

id	name	price	member
1	M5Stack	62.39	Yoshi
2	M5Stick C PLUS	29.8	Maxim
3	Quest Pro	1799.99	Maxim
4	Arduino Uno	24	Yoshi
5	Oculus Rift	400	Yoshi
6	Oscilloscope	299.99	Maxim
7	Ipad Pro 11 Inch	1149	Beat
8	Photocube	129	Beat
9	LEDs	0.2	Maxim
10	Voltage meter	40	Maxim

[←](#) DML Basics - 1

Code Editor



```
1 insert into items (name, price, member) VALUES
2 ("Macbook Pro 32GB", 2399, "Yoshi");
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
```

Question

WISE member Yoshi would like to add the item 'Macbook Pro 32GB' to the wishlist, the model costs 2599\$.

items
id: number (PK)
name: varchar
price: float
member: varchar

The WISE Wishlist database has only table, items.

In this items table the data is stored for all items that WISE lab members might want to buy in the near future for their research purposes. Every entry in the table contains an id for the item, the name of the item, the price of the item and the first name of the WISE member that wants to buy the item.

Checks

The table should have 19 elements



The WISE member should be Yoshi



The item name should be Macbook Pro 32GB



The price should be 2599



DDL Basics - 1

Code Editor



```
1 CREATE TABLE ticket_sales (
2     "id"      INTEGER NOT NULL,
3     "amount"   INTEGER NOT NULL UNIQUE,
4     "type"    TEXT NOT NULL,
5     "price"   REAL NOT NULL,
6     PRIMARY KEY("id" AUTOINCREMENT)
7 );
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
```

Question

Create the following db

ticket_sales

- id: integer PK auto-increment
- amount: integer
- type: text
- price: real

Checks

table ticket_sales exists



could insert two records



auto increment is working



TraceTables

The screenshot shows a web browser window titled "trace-tables.netlify.app" with the page header "Code Tracer". Below the header, the text "Select a trace type:" is displayed. There are four card-like sections, each representing a different trace type:

- Variable Steps** (Beginner):

Keep track of the steps that all the variables go through throughout the program.
Perfect for beginners.

[OPEN THE TABLE](#)
- Operators Table** (Beginner):

Basic trace table to log the use of the operators throughout the program's runtime.

[OPEN THE TABLE](#)
- Variable Values** (Intermediate):

Keep track of the values of all the variables throughout the program, a little more advanced.

[OPEN THE TABLE](#)
- Current Values** (Advanced):

Keep track of the current values in the program, recommended for advanced users.

[OPEN THE TABLE](#)

trace-tables.netlify.app

Variable steps

	line	name	action	value
<input type="checkbox"/>	3	⋮ x	declare, init	15
<input type="checkbox"/>	4	⋮ y	declare, init	25
<input type="checkbox"/>	5	⋮ x	read	15
<input type="checkbox"/>	5	⋮ y	read	25
<input checked="" type="checkbox"/>	5	⋮ area	declare, init	375

ADD STEP

trace-tables.netlify.app

Operators Table

Expression	Evaluates to
! ▾ isOk	true
x ++ ▾	8
x - ▾ y	5
isOk ? "yes" : "no"	"no"

REMOVE **ADD STEP**

trace-tables.netlify.app

Operators Table

Expression	Evaluates to
! ▾ isOk	true
x ++ ▾	8
x - ▾ y	5
isOk ? "yes" : "no"	"no"

Add Step

Unary-Prefix

Unary-Postfix

Binary

Ternary

trace-tables.netlify.app

Variable values

	Line number	Value	
x	3	15	<button>REMOVE LINE</button> <button>ADD LINE</button>
y	4	25	<button>REMOVE LINE</button> <button>ADD LINE</button>
area	5	375	<button>REMOVE LINE</button> <button>ADD LINE</button>

ADD VARIABLE

A screenshot of a web application titled "Current values" displayed in a browser window. The page shows a table of variables and their current values. The variables listed are x, y,isOk, name, and area. The values are 15, 25, checked, Yoshi, and 375 respectively. Each row has edit and delete icons. A green "ADD VARIABLE" button is at the bottom.

variable	value	
x	15	✖️ ⚡
y	25	✖️ ⚡
isOk	✓	✖️
name	Yoshi	✖️
area	375	✖️ ⚡

ADD VARIABLE

trace-tables.netlify.app

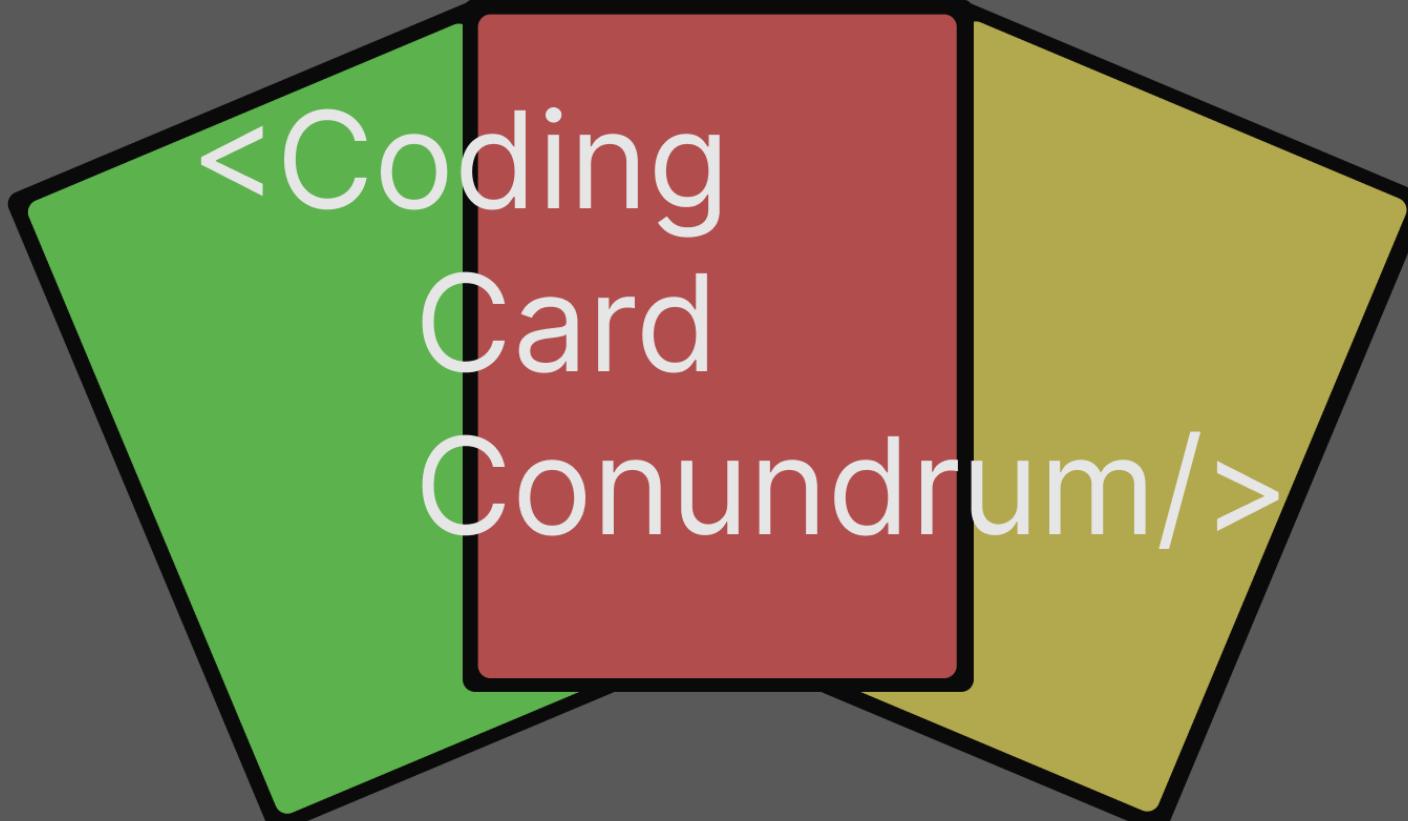
Current values

variable	value	
x	15	⬇️ ⚡
y	25	⬇️ ⚡
isOk	✓	⚡
name	Yoshi	⚡
area	375	⬇️ ⚡

Export trace

Print to PDF

Export as JSON



<Coding
Card
Conundrum/>

3 Types of Cards

Goal

```
assert(y < z &&
       z < x,
       "happy days :)");

```

- 2 -

Goal Cards

Environment

```
let foo = 65;
let bar = 90;
let x   = 43;
let y   = 31;
let z   = 28;

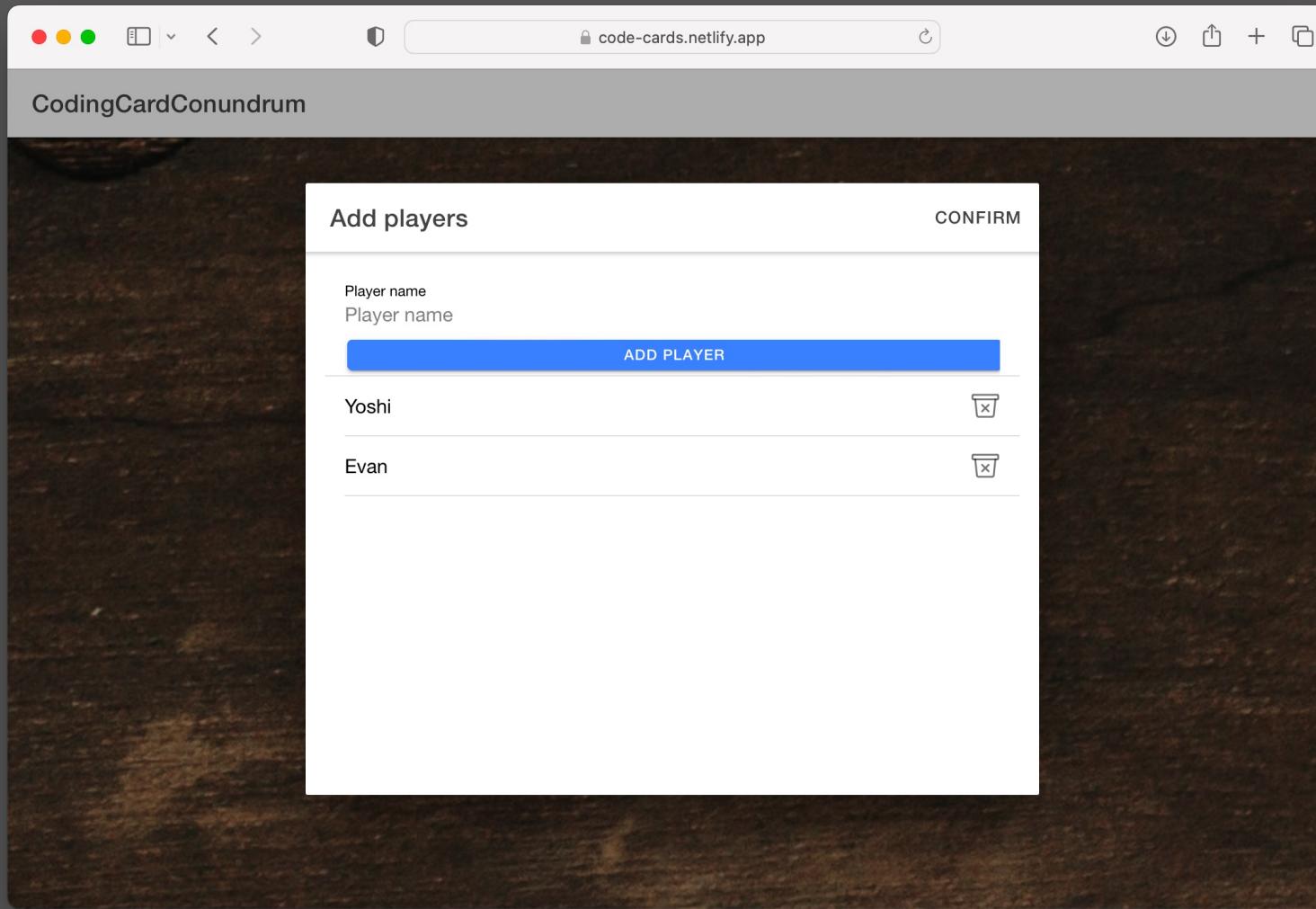
```

```
if (z >= bar) {
  z = z + 23;
  bar = bar - 96;
}
```

**Heavy
mathematics**

Code Cards

1– 4 Players



The Playing Field

The screenshot shows a web browser window with the URL `code-cards.netlify.app`. The title bar says "CodingCardConundrum - Yoshi's turn". The main content area is divided into three environments, each with a table of variable values and a code editor below it.

Environment 1 - 📁

foo	bar	x	y	z
6	74	38	16	63

```
let foo = 6;
let bar = 74;
let x = 38;
let y = 16;
let z = 63;
```

Environment 2 - 📁

foo	bar	x	y	z
50	15	15	76	7

```
let foo = 50;
let bar = 15;
let x = 15;
let y = 76;
let z = 7;
```

Environment 3 - 📁

foo	bar	x	y	z
51	61	44	44	75

```
let foo = 51;
let bar = 61;
let x = 44;
let y = 44;
let z = 75;
```

Goal

```
assert(foo < z &&
       z < x,
       "Happy days :)>");
```

Code Cards:

- Swap if bigger**:
if (bar > foo) {
 x = bar;
 bar = foo;
 foo = x;
}
- Heavy mathematics**:
if (foo >= y) {
 foo = foo % 54;
 y = y + 36;
}
- Heavy mathematics**:
if (x > z) {
 x = x % 58;
 z = z + 80;
}
- Heavy mathematics**:
if (bar !== foo) {
 bar = bar % 12;
 foo = foo - 81;
}
- Swap if smaller**:
if (foo < bar) {
 y = foo;
 foo = bar;
 bar = y;
}
- Goal**:
assert(foo < z &&
 z < x,
 "Happy days :)>");

- 2 -

Playing a Card

The screenshot shows a web browser window titled "CodingCardConundrum - Yoshi's turn". The interface is designed to look like a wooden board with cards. It features three environments (Environment 1, Environment 2, Environment 3) and a Goal card.

Environment 1 -
Variables:

foo	bar	x	y	z
6	74	38	16	63

Code:

```
let foo = 6;
let bar = 74;
let x = 38;
let y = 16;
let z = 63;
```


Action: ADD CARD HERE

Environment 2 -
Variables:

foo	bar	x	y	z
50	15	15	76	7

Code:

```
let foo = 50;
let bar = 15;
let x = 15;
let y = 76;
let z = 7;
```


Action: ADD CARD HERE

Environment 3 -
Variables:

foo	bar	x	y	z
51	61	44	44	75

Code:

```
let foo = 51;
let bar = 61;
let x = 44;
let y = 44;
let z = 75;
```


Action: ADD CARD HERE

Goal
Code:

```
assert(foo < z &&
z < x,
"happy days :)" );
```


Page number: - 2 -

Cards Available:

- Swap if bigger
- Heavy mathematics
- Heavy mathematics
- Heavy mathematics
- Swap if smaller

Updating the Trace Table

The screenshot shows a web application interface titled "CodingCardConundrum - Yoshi's turn". The interface is divided into three main sections, each representing an environment:

- Environment 1 -** Contains the initial state of variables:

foo	bar	x	y	z
6	74	38	16	63

Code:

```
let foo = 6;
let bar = 74;
let x = 38;
let y = 16;
let z = 63;
```

ADD CARD HERE
- Trace Table**: A central component for updating the trace table. It displays the current state of variables and provides an "ADD LINE" button to insert new rows into the table:

#	foo	bar	x	y	z
1	50	15			
2		15			
3			15		
4				76	
5					7

Code:

```
let foo = 50;
let bar = 15;
let x = 15;
let y = 76;
let z = 7;
```

CONFIRM
- Environment 3 -** Shows the state of variables after the trace table update:

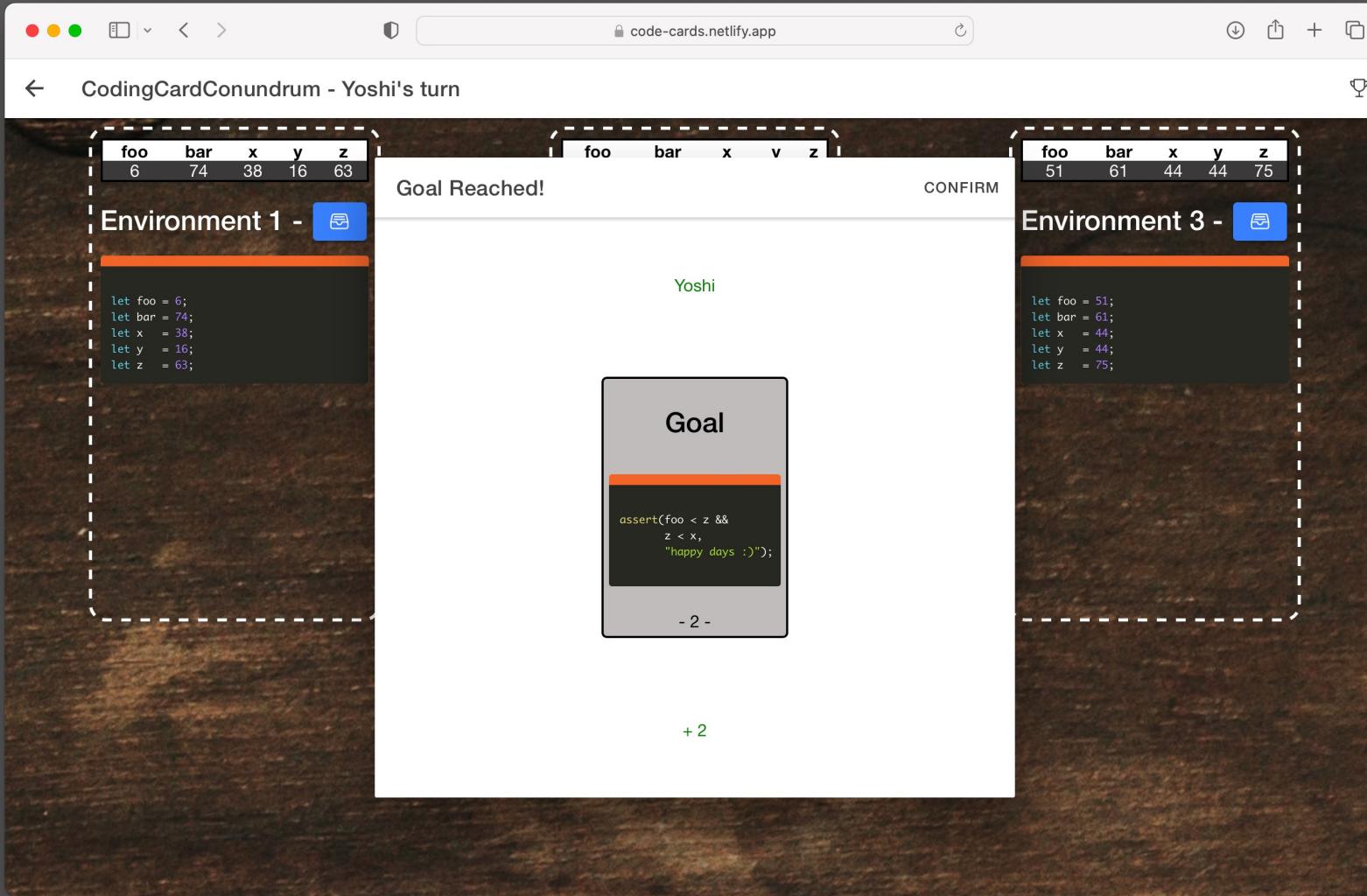
foo	bar	x	y	z
51	61	44	44	75

Code:

```
let foo = 51;
let bar = 61;
let x = 44;
let y = 44;
let z = 75;
```

ADD CARD HERE

Earning Points



KING'S SCROLL

The Search for the Chosen One



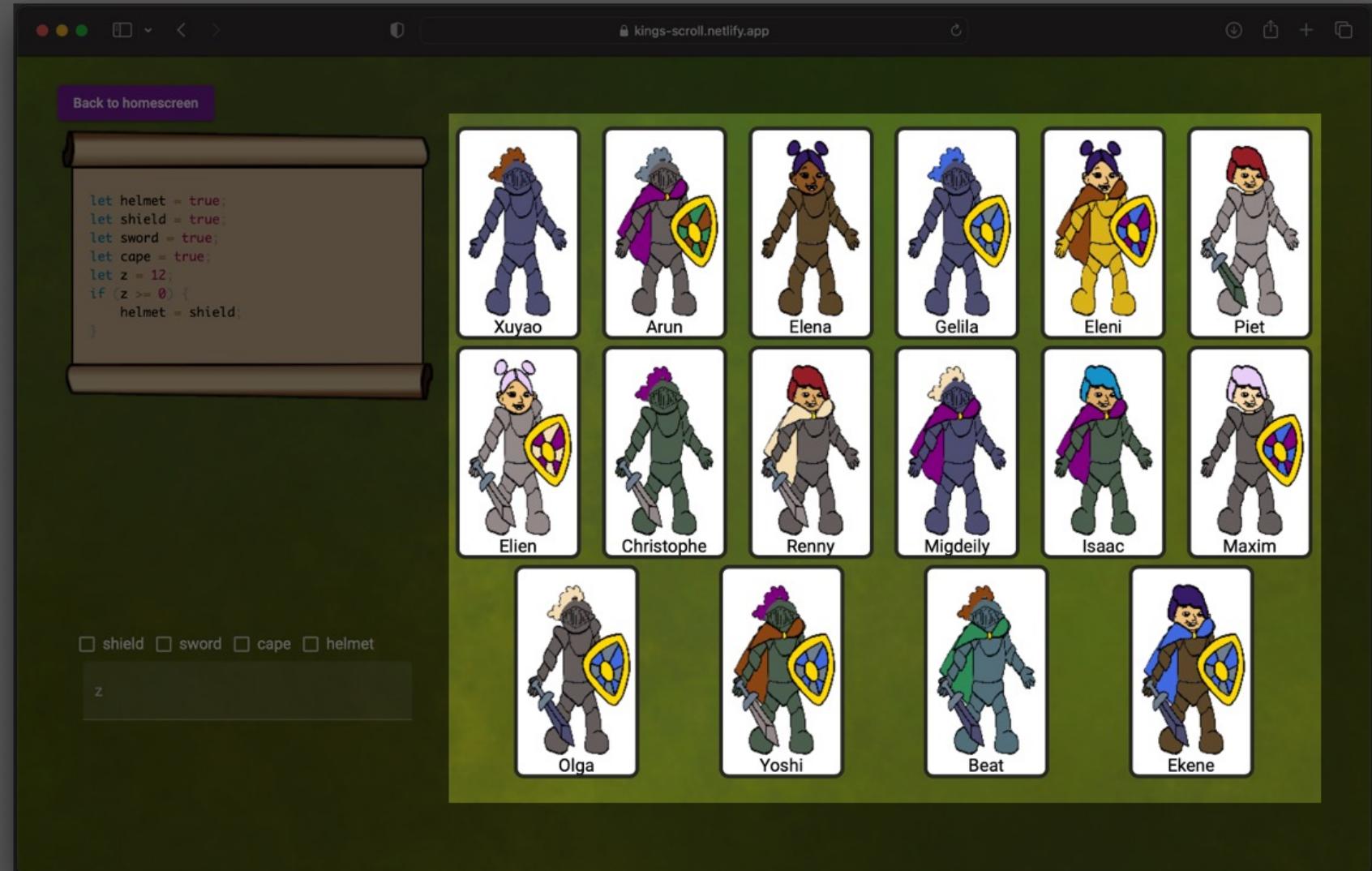
[Back to homescreen](#)

```
let helmet = true;
let shield = true;
let sword = true;
let cape = true;
let z = 12;
if (z >= 0) {
  helmet = shield;
}
```



Sixteen Heroes

- Shield
- Sword
- Helmet
- Cape
- Gender
- Skin tone



The Scroll

```
let helmet = true;
let shield = true;
let sword = false;
let cape = false;
let h = shield;
shield = cape;
cape = h;
let k = 6;
if (k < 13) {
    helmet = sword;
} else {
    shield = true;
}
```

```
let helmet = true;
let sword = true;
let shield = false;
let cape = false;
let p = cape;
cape = helmet;
helmet = p;
let q = 7;
do {
    q++;
    cape = !sword;
} while (q <= 18);
```

```
let cape = true;
let helmet = false;
let shield = false;
let sword = false;
function l() {
    sword = !cape;
}
cape = cape;
l();
let m = cape;
cape = shield;
shield = m;
```

```
let helmet = true;
let shield = false;
let sword = false;
let cape = false;
for (let d = 0; d < 12; d++) {
    cape = true;
}
```

The image shows a screenshot of a web browser window with a dark theme. The title bar reads "kings-scroll.netlify.app". The main content area has a dark green background.

Settings

Difficulty

Easy Medium Hard

Language features

- for loops
- if
- if ... else ...
- do while loops
- function declarations

[Back to main menu](#)

State Table

Back to homescreen

```
let helmet = true;
let shield = true;
let sword = true;
let cape = true;
let z = 12;
if (z >= 0) {
    helmet = shield;
}
```

shield sword cape helmet

z

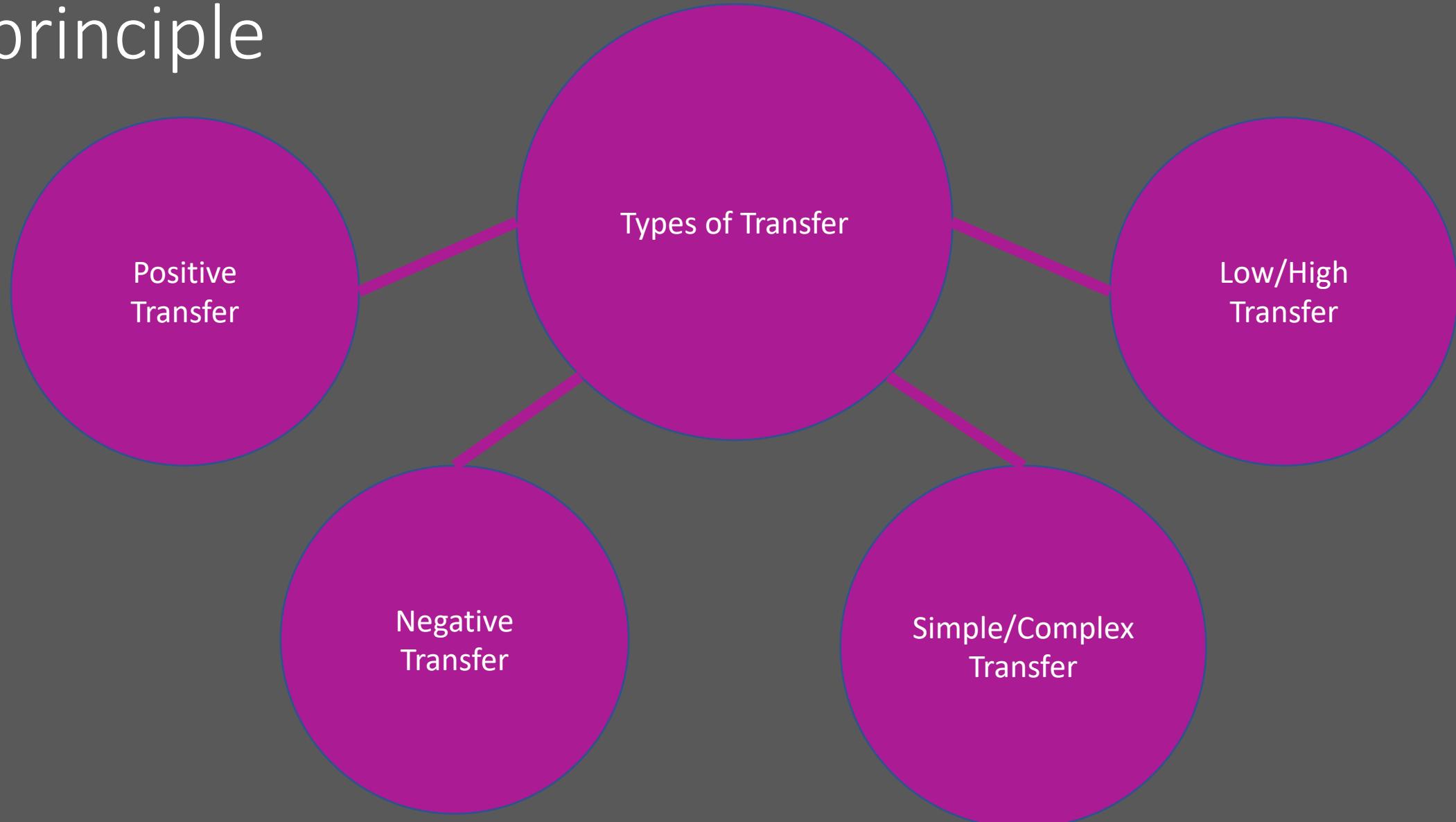
Xuyao	Arun	Elena	Gelila	Eleni	Piet
Elien	Christophe	Renny	Migdeily	Isaac	Maxim
Olga	Yoshi	Beat			Ekene

Guidelines

Embrace themes!



But don't forget about the skill transfer principle



Invite the social aspects!



Keep the setup minimal



Focus on one specific learning goal

Specific

Clear and specific outcomes?

Measurable

Define assessment/evaluation to measure outcomes?

Achievable

Is the expected level realistic?

Relevant

Is the goal relevant for the general goals of the learners?

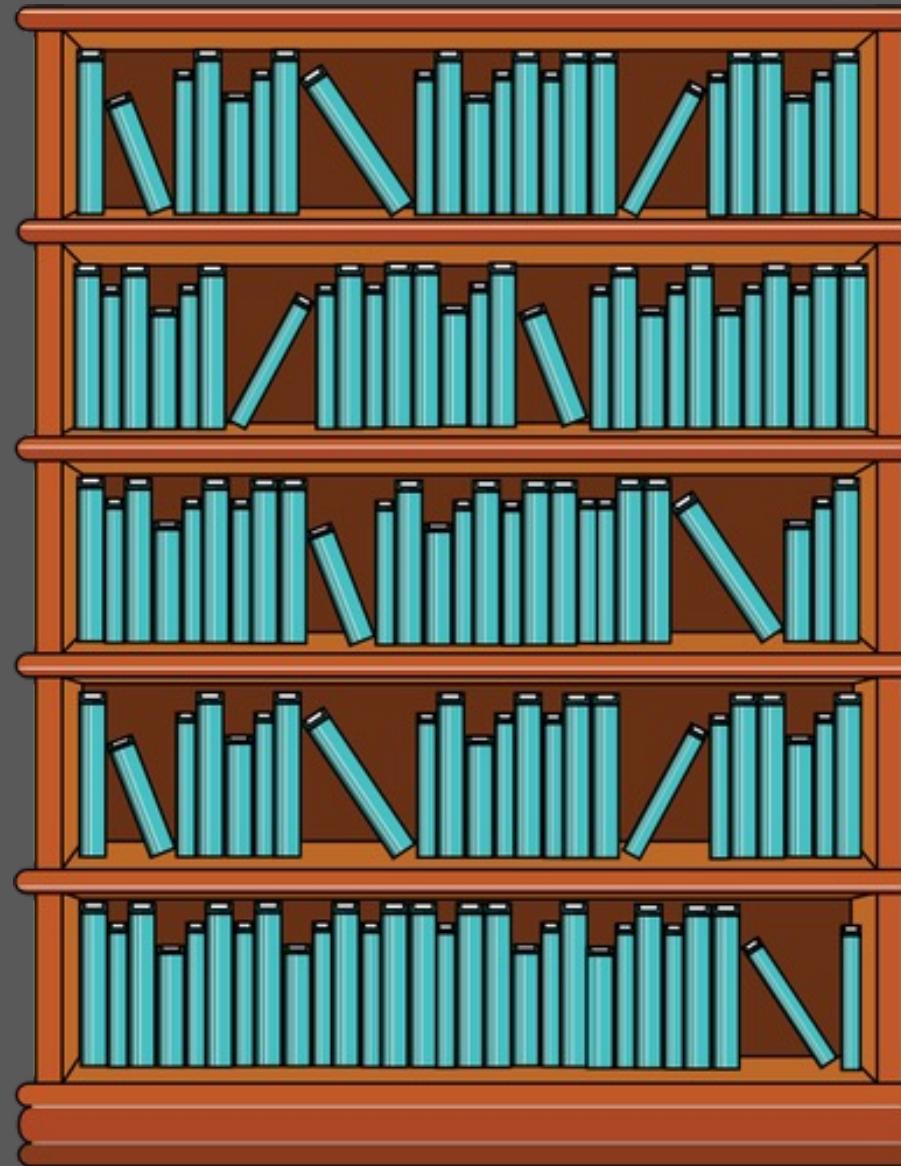
Time-limited

Can it be achieved in a logical time unit?

Keep in mind the expertise reversal principle



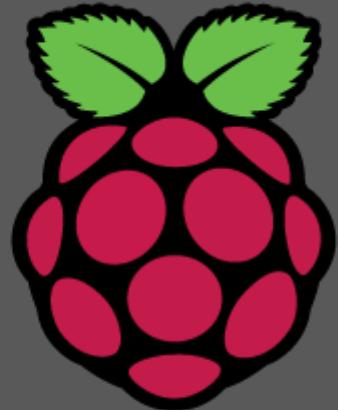
Automated content generation will be a lifesaver



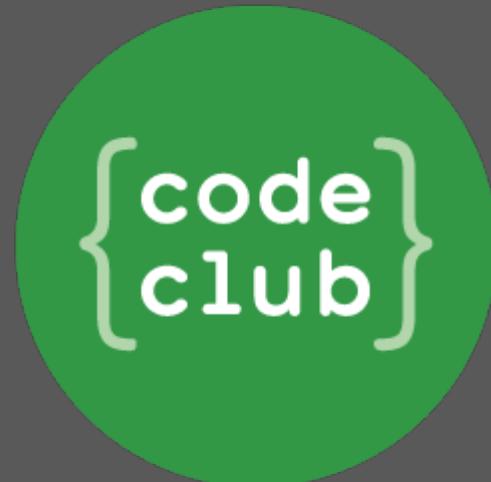
Make your things mobile compatible



Join us and build things



Raspberry Pi



CoderDojo

HACK
YO~~U~~UR
FUTURE

{ migra Code
..... }