cards against the liver

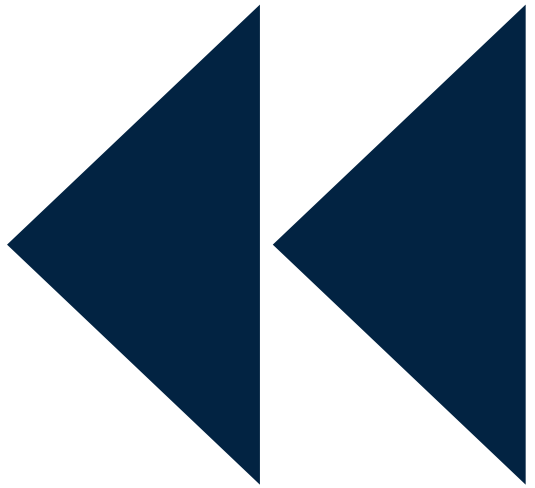
Projekt-Vorstellung Team C

Lasse Kreuter, Felix Wegener,
Maria Lorenz, Chiara Knipprath

Inhalt

- Rückblick
  - Projektidee
  - Umsetzung Design
  - Architektur
- Live Demo
- Technische Highlights
- Fazit

Rückblick



Projekt

- Kartenspiel App
  - Bettler
  - Schwimmen
  - Etc.
- Promillerechner

Material Design Farben

Hellgrau
#f9f9f9

Grau
#e4e9ed

Dunkeltürkis
#007d7c

Rot
#7d0000

Grün
#007d00

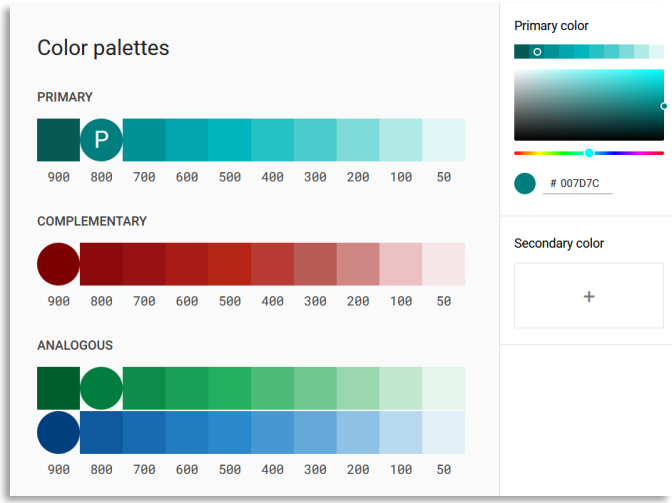
Dunkelblau
#003f7d

Türkis
#00a6ac

Rot
#7d0000

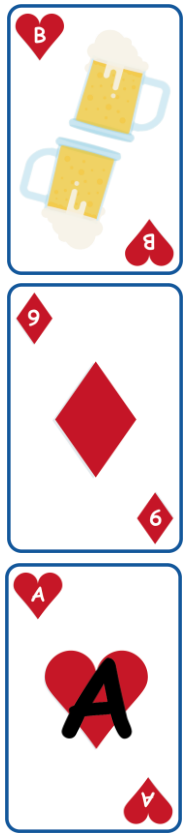
Grün
#007d00

Dunkelblau
#022342

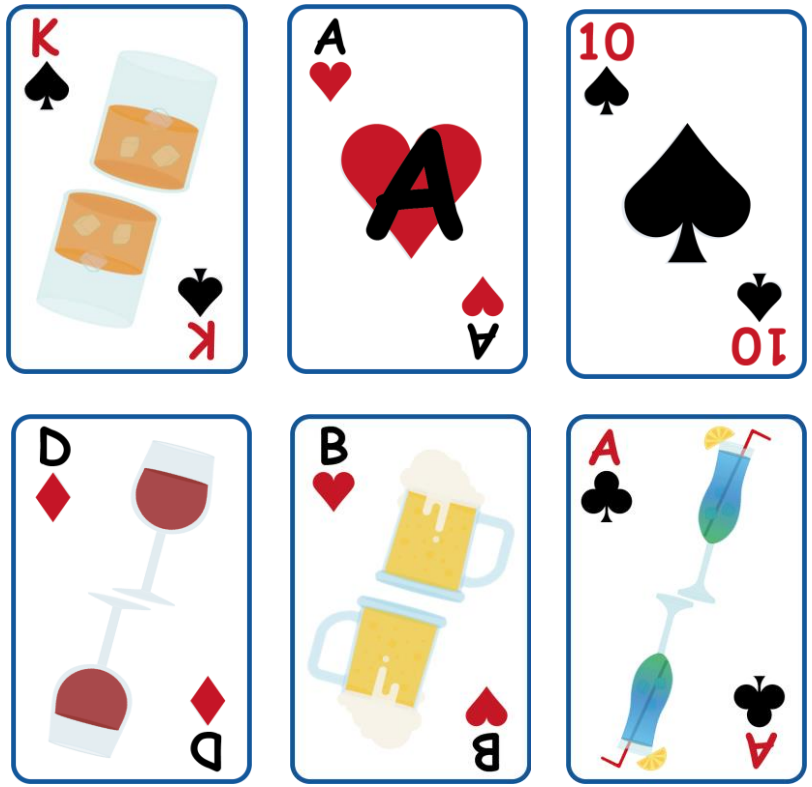


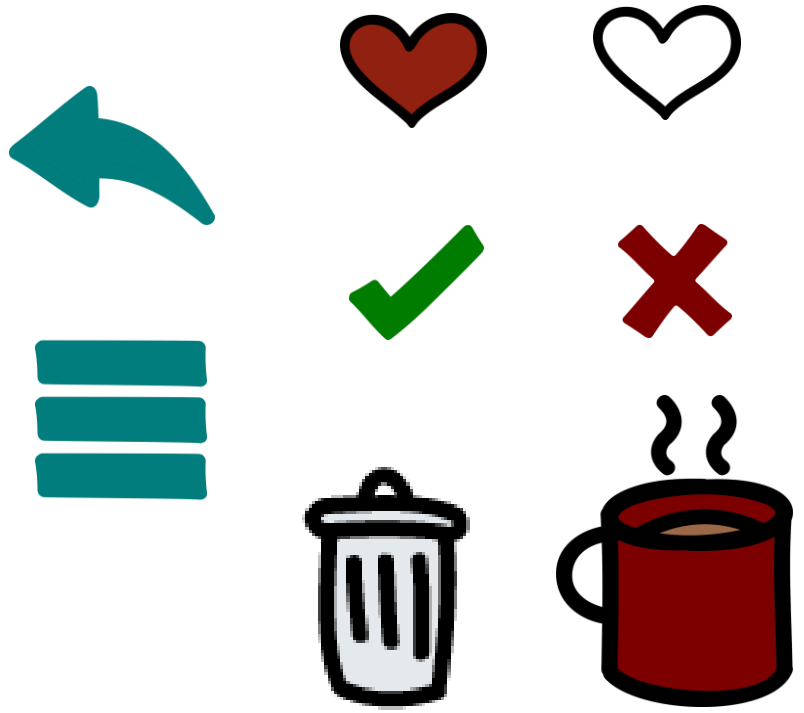
Design Karten & Icons

Alt



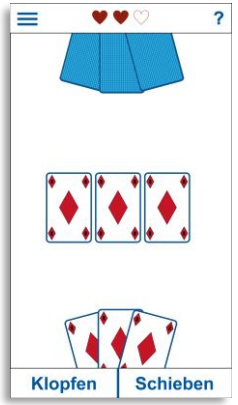
Neu



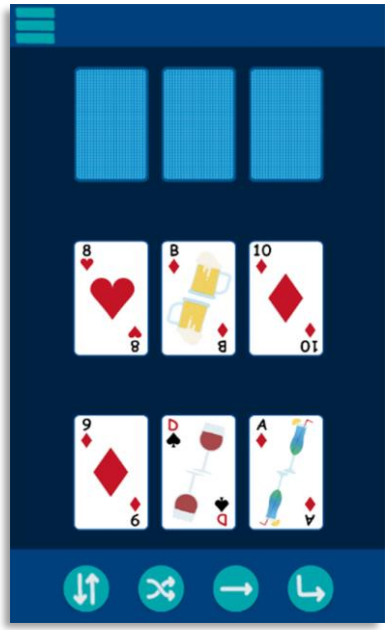


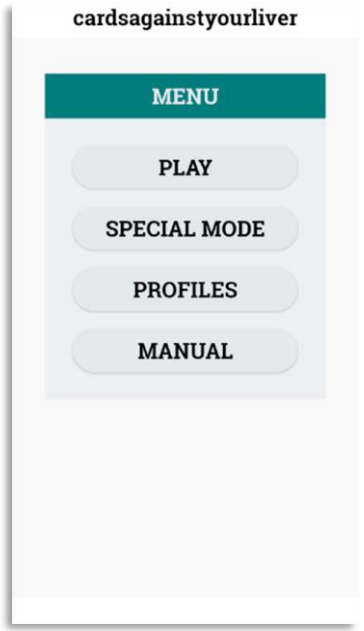
Finale Umsetzung

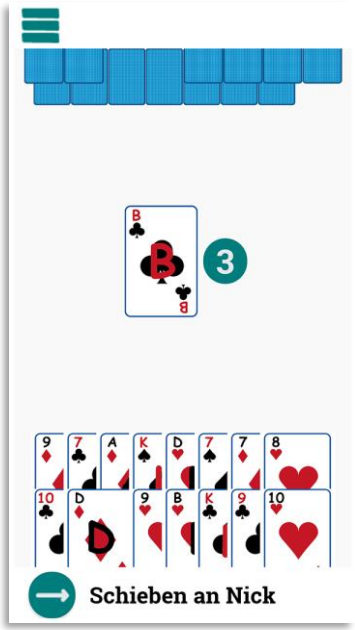
Alt



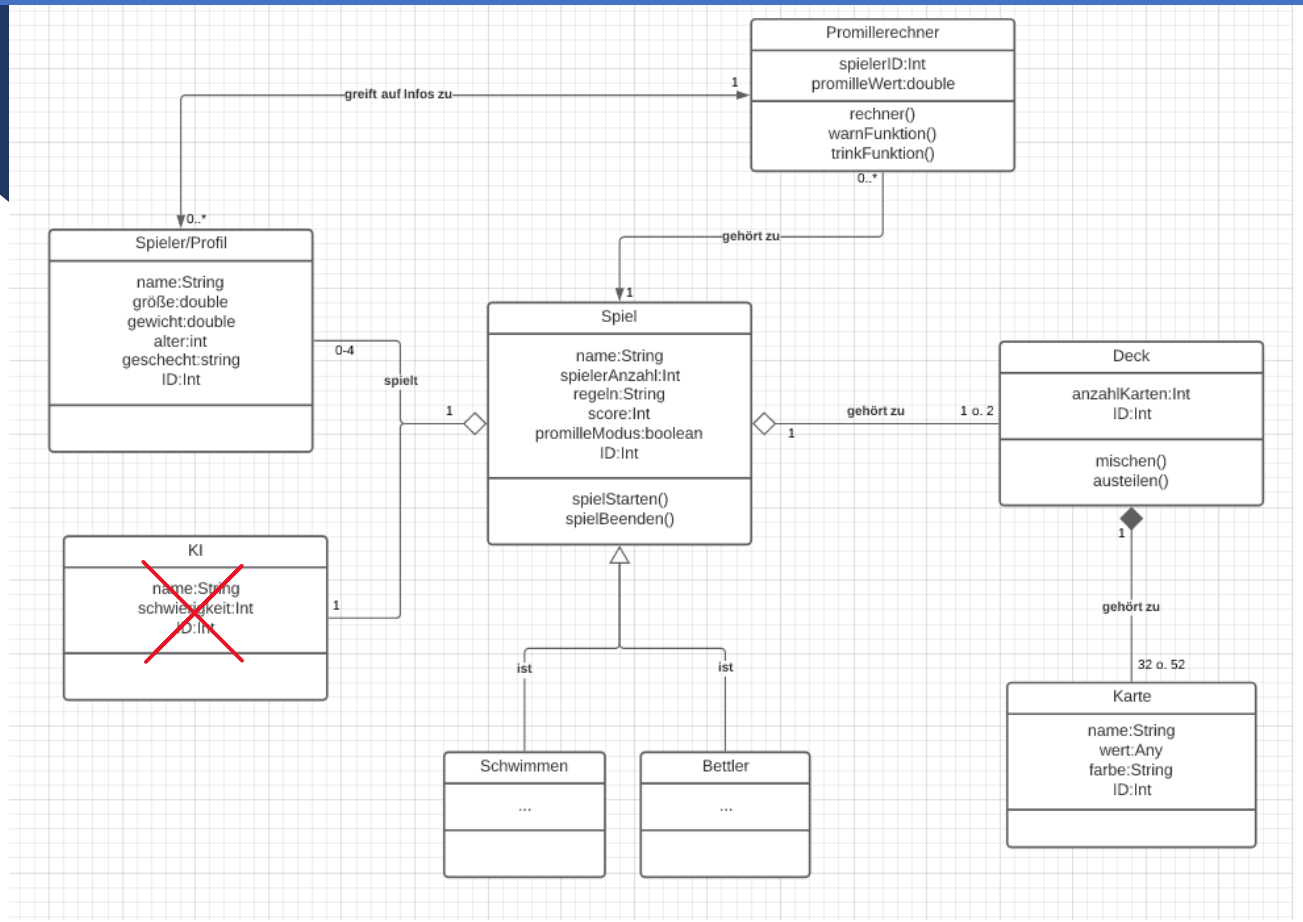
Neu

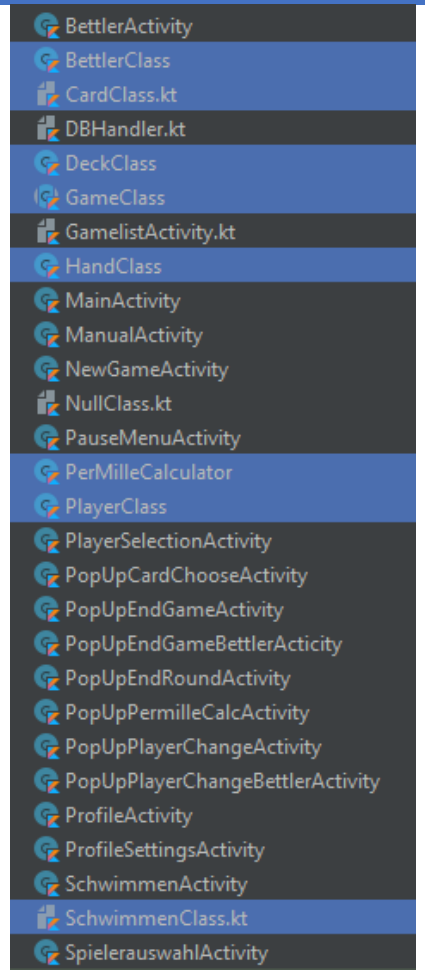






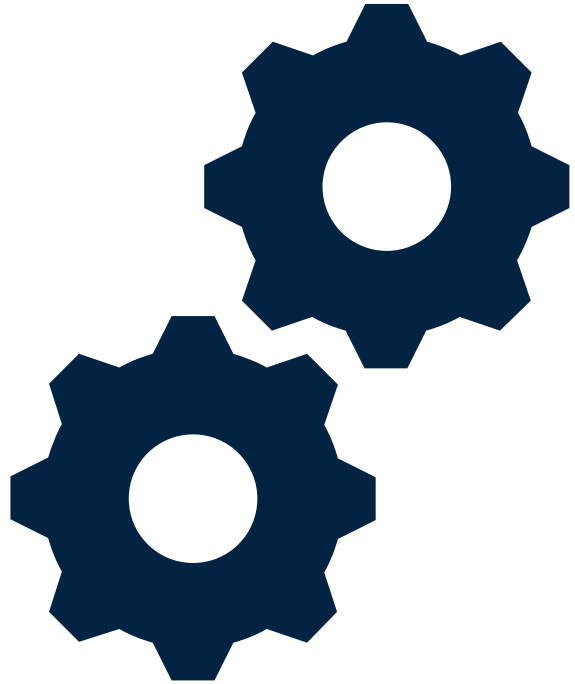
Architektur





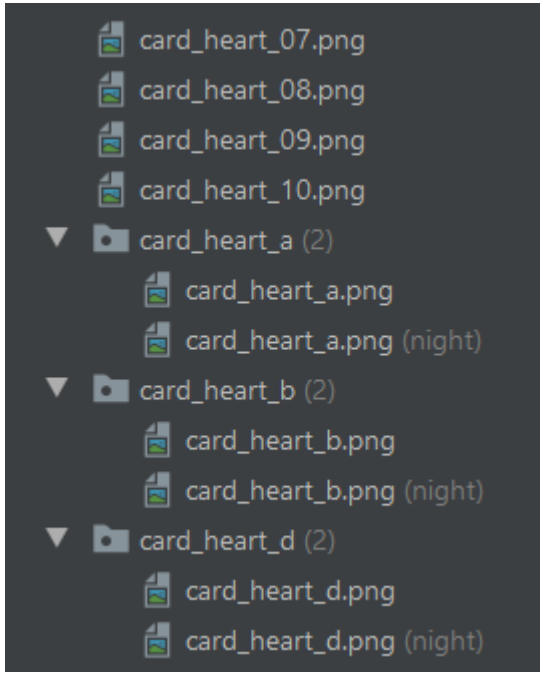
Live Demo

Technische Highlights



Technische Highlights

- Spielmodus Wechsel




```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)

    val appSettingPrefs: SharedPreferences = getSharedPreferences( name: "AppSettingPrefs",  mode: 0)
    val sharedPrefsEdit: SharedPreferences.Editor = appSettingPrefs.edit()
    val isNightModeOn: Boolean = appSettingPrefs.getBoolean( key: "NightMode",  defValue: false)

    if (isNightModeOn) {
        AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.MODE_NIGHT_YES)
    } else {
        AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.MODE_NIGHT_NO)
    }


    specialmode_button.setOnClickListener(View.OnClickListener { it: View!
        if (isNightModeOn) {
            AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.MODE_NIGHT_NO)
            sharedPrefsEdit.putBoolean("NightMode", false)
            sharedPrefsEdit.apply()
        } else {
            AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.MODE_NIGHT_YES)
            sharedPrefsEdit.putBoolean("NightMode", true)
            sharedPrefsEdit.apply()
        }

    })
}
```

Technische Highlights

- Spielkarten Logik

```
//Dies repräsentiert eine Spielkarte, sie hat ein Sign(Pik,Herz...) und ein Value (2,3,Bube...)
class CardClass(private val sign: Sign, private val value: Value) : Null() {
```

```
override fun getPic(): Int {
    var picture: Int = R.drawable.card_kreuz_07
    when (sign) {
        Sign.HERZ -> {
            when (value) {
                Value.SIEBEN -> picture = R.drawable.card_heart_07
                Value.ACHT -> picture = R.drawable.card_heart_08
                Value.NEUN -> picture = R.drawable.card_heart_09
                Value.ZEHN -> picture = R.drawable.card_heart_10
                Value.BUBE -> picture = R.drawable.card_heart_b
                Value.DAME -> picture = R.drawable.card_heart_d
                Value.KOENIG -> picture = R.drawable.card_heart_k
                Value.ASS -> picture = R.drawable.card_heart_a
```

```
//das Value in einer Int Zahl
override fun getValueNumber(): Int {
    var valueNumber: Int = 0
    when (value) {
        Value.ZWEI -> valueNumber = 2
        Value.DREI -> valueNumber = 3
        Value.VIER -> valueNumber = 4
        Value.FUENF -> valueNumber = 5
        Value.SECHS -> valueNumber = 6
```

```
enum class Sign {
    HERZ, KARO, PIK, KREUZ
}


enum class Value {
    ZWEI, DREI, VIER, FUENF, SECHS, SIEBEN, ACHT,
    NEUN, ZEHN, BUBE, DAME, KOENIG, ASS
}
```

Technische Highlights

- Spielkarten Logik

```
var deck: MutableList<Null> = mutableListOf()

init {
    if (deckSize == 1) {
        for (sign in Sign.values())
            for (value in Value.values())
                deck.add(CardClass(sign, value))

    } else if (deckSize == 2) {
        for (sign in Sign.values())
            for (value in Value.values())
                if (value != Value.ZWEI && value != Value.DREI && value != Value.VIER && value != Value.FUENF && value != Value.SECHS) {
                    deck.add(CardClass(sign, value))
                }
    }
```

```
fun shuffle() = deck.shuffle()

//eine Karte wird aus dem Deck gezogen, wenn deck leer kommt Null
fun drawCard(): Null {
    if (deck.isNotEmpty())
        return deck.removeAt( index: 0)
    else
        return Null()
}
```

Fazit

Fazit - Was funktioniert gut?

- Kartenspiele
- Übersetzung der Mockups
- PopUp Menüs
- Datenbank
- Promillerechner

Fazit - Was wurde evtl. nicht ganz fertig?

- Drag & Drop mit Spielen verbinden
- KI

Fazit - Was war schwerer als erwartet?

- Layout Bettler
- Drag & Drop mit Spielen verbinden
- Spiele in Android mit Bildern + PopUps verbinden