

Dokumentasi Project Ujian Tengah Semester (UTS)  
Pemrograman Nirkabel  
Genap 2019/ 2020

Dikerjakan Oleh:  
Varianto (160717023)

activity\_main.xml



Masukkan kata (max 8 karakter)

Masukkan kata sinonimnya (max 8 karakter)

Untuk tampilan input dengan border seperti ini, pertama-tama saya membuat editText\_border.xml di folder app > res > drawable. Isi dari xml tersebut adalah:

```
...  
    android:shape="rectangle"  
    <stroke  
        android:color="@color/colorPrimary"  
        android:width="2dp" />  
...
```

Saya set dengan  
nilai #808080

Kemudian saya menggunakan isi xml tersebut sebagai background editText, pada properti background: @drawable/edittext\_border.

MainActivity, PlayerStartActivity, ResultActivity

```
...  
class MainActivity : AppCompatActivity() {  
    override fun onBackPressed() {  
        // super.onBackPressed();  
        // Not calling **super**, disables back button in  
        current screen.  
    }  
    ...  
}
```

Codingan Handle Back Button ini saya tambahkan di seluruh activity untuk mencegah user kembali ke halaman sebelumnya apabila user menekan tombol back pada device

```
override fun onCreate(savedInstanceState:  
Bundle?) {  
    super.onCreate(savedInstanceState)  
    requestWindowFeature(Window.FEATURE_N  
O_TITLE)  
    window.addFlags(WindowManager.LayoutPar  
ams.FLAG_FULLSCREEN)  
    supportActionBar?.hide()  
    ...  
}
```

Codingan ini membuat tampilan layar aplikasi full screen ketika seluruh activity diakses oleh device pemain

## MainActivity

...

```
onCreate(savedInstanceState: Bundle?) {
```

...

```
    var currId = this.intent.getStringExtra("id")
```

```
    var id = if(!currId.isNullOrEmpty()) currId else "1"
```

```
    var playersScore = this.intent.getStringExtra("playersScore")
```

```
    textViewPlayerQuestion.setText("Soal Untuk Player ${id}")
```

```
    textViewGivePhoneToPlayer.setText("Berikan HP ini ke Player ${id}")
```

```
    buttonStartPlayer.setText("Mulai Pemain ${id}")
```

...

```
}
```

Mendapatkan nilai id pemain saat ini dari activity lainnya, bisa bernilai "1" atau "2". Bernilai "1" jika hasil permainan telah ditampilkan dan new game dilakukan. Bernilai "2" jika giliran pemain berganti ke Player 2.

Menampung nilai currId ke variabel bernama id.  
Var ini akan digunakan seterusnya

Mendapatkan score pemain terakhir

Menset nilai pada TextView sesuai dengan giliran pemain "1" atau "2" yang sedang bermain

## MainActivity (lanjutan)

```
...
buttonStartPlayer.setOnClickListener {
    var isError = false
    val kata = editTextKata.text.toString()
    val sinonim = editTextSinonim.text.toString()

    // check character
    if(kata.isEmpty()) { isError = true; alert(Appcompat, "Isikan field kata").show() }
    if(sinonim.isEmpty()) { isError = true; alert(Appcompat, "Isikan field sinonim").show() }

    if(kata.length > 8) { isError = true; alert(Appcompat, "Panjang kata maximal 8 karakter").show() }
    if(sinonim.length > 8) { isError = true; alert(Appcompat, "Panjang sinonim maximal 8 karakter").show() }

    if(!kata.matches(Regex("^([a-zA-Z]*$)"))) { isError = true; alert(Appcompat, "Field kata harus merupakan alfabet saja").show() }
    if(!sinonim.matches(Regex("^([a-zA-Z]*$)"))) { isError = true; alert(Appcompat, "Field sinonim harus merupakan alfabet
saja").show() }
}
...
```

Membuat variabel penampung kondisi aplikasi yang diperlukan pada MainActivity.

isError mengecek apakah ada kesalahan dalam penginputan kata; kata adalah nilai dari editTextKata; sinonim adalah nilai dari editTextSinonim

Alert user jika editText kata atau sinonim dikosongkan atau karakter berjumlah lebih dari 8 huruf

Cek apakah input user berupa alfabet atau tidak dengan menggunakan Regular Expression. Alert user jika terdapat input berupa angka

## MainActivity (lanjutan)

```
...
buttonStartPlayer.setOnClickListener {
    ...
    if(!isError) {
        startActivity<PlayerStartActivity>(
            "id" to id,
            "kata" to kata.toUpperCase(),
            "sinonim" to sinonim.toUpperCase(),
            "playersScore" to if(!playersScore.isNullOrBlank()) playersScore else ""
        )
    }
}
...
```

Jika semua inputan sesuai, saya menggunakan Intent untuk memindahkan user ke PlayerStartActivity dengan data id pemain sebelumnya, kata, sinonim, dan skor pemain sebelumnya, bernilai "" (kosong) jika game baru dimainkan.

## PlayerStartActivity

```
...  
fun generateCorrectAnswer(sinonim:String, numOfCorrect:Int) {  
    // remove all existing components  
    TableRowCorrectAnswer.removeAllViewsInLayout()  
  
    for(i in 1..sinonim.length) {  
        var editText: EditText  
        editText = EditText(this)  
        TableRowCorrectAnswer.addView(editText)  
  
        editText.setText(if(i <= numOfCorrect) "${sinonim[i-1]}" else " ")  
        editText.gravity = Gravity.CENTER  
        editText.isEnabled = false  
        editText.isFocusable = false  
        editText.layoutParams.width = 44  
    }  
}
```

```
override fun onCreate(savedInstanceState: Bundle?) { ... }
```

Menampilkan jawaban (sinonim) yang dijawab benar pada activity\_player\_start.xml. Parameter sinonim adalah kata sinonim yang akan ditebak. Parameter numOfCorrect adalah berapa banyak huruf yang telah berhasil ditebak pemain.

Setiap kali fungsi dipanggil, seluruh karakter pada sinonim akan di-loop dan fungsi akan membuat editText dengan karakter-karakter yang terjawab benar oleh pemain.

Contoh: generateCorrectAnswer("DONASI", 5) akan menampilkan hasil berikut.



## PlayerStartActivity

```
...  
fun getRandomString(length: Int) : String {  
    val allowedChars = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"  
    return (1..length)  
        .map { allowedChars.random() }  
        .joinToString("")  
}
```

Fungsi akan dipanggil pertama kali `activity_player_start.xml` berjalan untuk menampilkan 18 huruf secara acak. Parameter `length` pada fungsi menunjukkan berapa banyak karakter yang akan dihasilkan sebagai output dari fungsi.

```
override fun onCreate(savedInstanceState: Bundle?) {  
    val id = this.intent.getStringExtra("id")  
    val kata = this.intent.getStringExtra("kata")  
    val sinonim = this.intent.getStringExtra("sinonim")  
    var sinonimList = listOf<String>().toMutableList()  
    var playersScore = this.intent.getStringExtra("playersScore")
```

Mendapatkan nilai intent yang dikirimkan dari `MainActivity` dan ditampung ke variabel-variabel terkait. Var `sinonimList` adalah list yang anggotanya adalah seluruh karakter dari sinonim yang akan diisi pada ekspresi `for` di bawahnya beserta dengan karakter random sisa.

```
    for (i in 1..sinonim.length) { sinonimList.add(sinonim[i-1].toString()) }
```

```
    var numOfCorrect = 0  
    var score = 100  
    textViewSkor.text = "Skor: ${score}"  
    textViewKata.text = kata
```

Menyiapkan variabel `numOfCorrect` untuk melacak berapa banyak karakter yang telah ditebak pemain dengan benar, serta skor pemain (nilai default awal adalah 100). Kemudian, skor dan kata untuk ditebak akan ditampilkan ke `activity_player_start.xml`

```
    ...  
}
```



## PlayerStartActivity (lanjutan)

...

```
generateCorrectAnswer(sinonim, 0)
```

Mengenerate karakter-karakter yang terjawab benar pada ctivity\_player\_start.xml

```
while(sinonimList.size < 18) {  
    var randomString = getRandomString(1)  
    if(!sinonimList.contains(randomString)) {  
        sinonimList.add(randomString)  
    }  
}
```

Mengisikan var sinonimList dengan karakter random sisa hingga berisi total 18 karakter. Agar memastikan tidak ada duplikasi karakter, maka ada pengecekan apakah di sinonimList sudah ada karakter tertentu

```
val shuffledSinonimList = sinonimList.shuffled()
```




```
for(i in 1..18) {  
    var button: Button  
    button = Button(this)  
  
    if(i in 1..6) { TableRow1Random.addView(button) }  
    if(i in 7..12) { TableRow2Random.addView(button) }  
    if(i in 13..18) { TableRow3Random.addView(button) }
```

```
    button.setText("${shuffledSinonimList.elementAt(i-1)}")  
    button.layoutParams.width = 60  
}
```

...

```
}
```

Menghasilkan 18 kotak sebagai button-button penampung karakter random. Namun terlebih dahulu saya membuat layout Tabel dengan 3 buah Row yang kosong seperti berikut.

▼  TableRow1Random  
 TableRow2Random  
 TableRow3Random

Button-button dimasukkan ke dalam 3 bagian baris sesuai dengan index masing-masing dengan ilustrasi berikut.

H	V	Q	A	G	B	index 1-6
T	U	N	N	C	Y	index 7-12
R	W	J	A	P	Z	index 13-18

## PlayerStartActivity (lanjutan)

```
...
button.setOnClickListener {
    if(numOfCorrect != sinonim.length) {
        val selectedText = (it as Button).text.toString()

        if (selectedText == sinonim[numOfCorrect].toString()) {
            numOfCorrect += 1
            generateCorrectAnswer(sinonim, numOfCorrect)

            it.setBackgroundColor(Color.GREEN)
            it.isEnabled = false

            if(numOfCorrect == sinonim.length) {
                if(id == "1") { playersScore = score.toString() }
                if(id == "2") { playersScore += "&${score.toString()}"; buttonNextPlayer.setText("RESULTS") }
                buttonNextPlayer.isVisible = true
            }
        }
        else {
            score -= 10
            textViewSkor.text = "Skor: ${score}"
        }
    }
}
...
```

Ketika button yang berisi sebuah karakter diklik, program akan mengecek apakah karakter yang diklik tersebut merupakan urutan karakter yang benar. Jika tidak, maka skor pemain akan berkurang 10.

Jika karakter yang ditebak benar, maka program akan menjalankan func `generateCorrectAnswer`, membuat background button penampun berwarna hijau. Dan, jika seluruh karakter telah tertebak dengan benar, maka button dengan tulisan "Next Player" atau "Results" akan ditampilkan

Var `playersScore` menampung nilai skor dari pemain 1 dan 2 dengan format "skor1&skor2", contohnya "80&40". Jika yang bermain saat ini masih pemain 1, maka `playersScore` hanya akan berisi "skor1", misalnya "80".

## PlayerStartActivity (lanjutan)

```
...  
buttonNextPlayer.setOnClickListener {  
    if(id == "1") {  
        // Direct user to MainActivity for inputting questions to player 2  
        startActivity<MainActivity>(  
            "id" to "2",  
            "playersScore" to playersScore  
        )  
    }  
    else {  
        // Direct user to ResultActivity  
        startActivity<ResultActivity>(  
            "playersScore" to playersScore  
        )  
    }  
}  
...
```

Ketika buttonNextPlayer ditekan, program akan mengecek apakah player saat ini adalah pemain 1 atau pemain 2. Jika pemain 1, maka program akan memindahkan user ke MainActivity untuk berganti giliran bermain

Jika pemain kedua telah selesai bermain (pemain 1 dan pemain 2 telah bermain), maka program akan diarahkan ke ResultActivity

## ResultActivity

```
...  
val playersScore = this.intent.getStringExtra("playersScore")  
var playersScoreArr = playersScore.split("&").toArray()  
var score1 = playersScoreArr[0]  
var score2 = playersScoreArr[1]
```

Pada ResultActivity, pertama-tama program akan mendapatkan skor masing-masing player dari intent playersScore. Skor masing2 pemain dengan format "skor1&skor2" akan dipisah dengan .split("&"), karena "&" merupakan penghubung. Hasil split dijadikan ke bentuk array. Skor pemain 1 berada di index 0, sebaliknya skor pemain 2 berada di index 1

```
textViewScorePlayer1.setText(score1)  
textViewScorePlayer2.setText(score2)
```

Skor masing-masing pemain akan ditampilkan ke textView

```
buttonNewGame.setOnClickListener {  
    startActivity<MainActivity>(  
        "id" to "1",  
        "playersScore" to ""  
    )  
}  
}  
...
```

Ketika tombol "New Game" diklik, maka program akan mengembalikan game ke MainActivity dengan nilai intent seperti game dalam keadaan semula, yaitu id bernilai "1" dan "playersScore" kosong = ""