Android Studio - JetPack Compose

Ogólne

Aplikacja prezentuje podstawy kompozycji elementów w oparciu o język Kotlin i JetPack Compose. Wersja rozbudowana zawiera dodatkowe elementy.

Uwaga: Nigdy nie wolno kopiować kodu z PDF-a, gdyż zawiera on niewidoczne znaki końca linii i tzw. twarde odstępy. Kod należy przepisać ze zrozumieniem.

Kod - wersja podstawowa

```
@file:OptIn(ExperimentalMaterial3Api::class)
package com.example.nazwaProjektu

import android.os.Bundle
import android.widget.Toast
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.compose.foundation.background
import androidx.compose.foundation.layout.Arrangement
import androidx.compose.foundation.layout.Box
import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.Row
import androidx.compose.foundation.layout.Spacer
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.fillMaxWidth
import androidx.compose.foundation.layout.height
import androidx.compose.foundation.layout.padding
import androidx.compose.foundation.layout.size
import androidx.compose.foundation.rememberScrollState
import androidx.compose.foundation.verticalScroll
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.Menu
import androidx.compose.material3.Divider
import androidx.compose.material3.ExperimentalMaterial3Api
import androidx.compose.material3.Icon
import androidx.compose.material3.IconButton
import androidx.compose.material3.MaterialTheme
import androidx.compose.material3.Surface
import androidx.compose.material3.Text
import androidx.compose.material3.TopAppBar
import androidx.compose.material3.TopAppBarDefaults
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
```

```
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.platform.LocalContext
import androidx.compose.ui.text.font.FontFamily
import androidx.compose.ui.text.font.FontStyle
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
import com.example.surname.ui.theme.SurnameTheme


class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            NazwaProjektuTheme { // Nazwa tego elementu zależy od nazwy projektu
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colorScheme.background
                ) {
                    MojeElementy()
                }
            }
        }
    }
}

@Composable
fun MojeElementy() {
    val contextForToast = LocalContext.current.applicationContext
    Column(
        horizontalAlignment = Alignment.CenterHorizontally,
        modifier = Modifier
            .background(Color.DarkGray)
            .fillMaxWidth()
            .fillMaxWidth()
            .padding(20.dp)) {
        TopAppBar(
            title = { Text("Nazwa aplikacji")},
            navigationIcon = {
                IconButton(onClick = {
                  Toast.makeText(contextForToast, "Naciśnięto ikonę nawigacji.",
                        Toast.LENGTH_SHORT).show()
                }) {
                    Icon(imageVector = Icons.Default.Menu, contentDescription = "Menu")
                }
            }
        )
```

```
        Text("Hello World! Welcome to Tychy!", color = Color.White)

        Text("Druga linia tekstu.", color = Color.White)

        Row(horizontalArrangement = Arrangement.End,
            verticalAlignment = Alignment.CenterVertically,
            modifier = Modifier
                .fillMaxWidth()
                .padding(10.dp)
                .background(Color.Gray)) {
            Text("EL1", fontWeight = FontWeight.Bold, modifier = Modifier.padding(10.dp))
            Text("EL2", fontWeight = FontWeight.Bold, modifier = Modifier.padding(10.dp))
            Box(
                modifier = Modifier
                    .size(100.dp)
                    .padding(10.dp)
                    .background(Color.Yellow),
                contentAlignment = Alignment.Center
            ) {
                Text("EL3")
            }
            Box(
                modifier = Modifier
                    .size(100.dp)
                    .padding(10.dp)
                    .background(Color.Yellow),
                contentAlignment = Alignment.Center
            ) {
                Text("EL4")
            }
        }

        Text("twoje Imię i Nazwisko",
            color = Color.Blue,
            fontSize = 30.sp,
            fontFamily = FontFamily.Cursive,
            modifier = Modifier
                .background(Color.LightGray)
                .padding(all = 20.dp))
    }
}
```

Kod - wersja rozbudowana

```
@file:OptIn(ExperimentalMaterial3Api::class)
package com.example.nazwaProjektu
```

```
// Można importować pojedyncze klasy lub zbiorczo całe moduły za pomocą gwiazdki:
// import androidx.compose.ui.text.font.FontStyle
// import androidx.compose.ui.text.font.FontWeight
// import androidx.compose.ui.text.font.FontFamily
// import androidx.compose.ui.text.font.*
import android.os.Bundle
import android.widget.Toast
// ...
// Inne moduły...


class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            NazwaProjektuTheme { // Nazwa tego elementu zależy od nazwy projektu
                // A surface container using the 'background' color from the theme
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colorScheme.background
                ) {
                    MojeElementy()
                    // Toast wewnątrz MainActivity:
                    val wiadomosc = "Hello Kity!"
                    val czasWyswietlania = Toast.LENGTH_LONG
                   val mojToast = Toast.makeText(applicationContext, wiadomosc, czasWyswietlania)
                    mojToast.show()
                }
            }
        }
    }
}


@Composable
fun MojeElementy() {
    val contextForToast = LocalContext.current.applicationContext
    Column(
        horizontalAlignment = Alignment.CenterHorizontally,
        modifier = Modifier
            .background(Color.DarkGray)
            .fillMaxWidth()
            .fillMaxWidth()
            // Możliwość przewijania kolumny, która nie mieści się na ekranie:
            .verticalScroll(rememberScrollState())
            .padding(20.dp)) {
        TopAppBar(
            title = { Text("Nazwa aplikacji")},
```

```
    navigationIcon = {
        IconButton(onClick = {
          Toast.makeText(contextForToast, "Naciśnięto ikonę nawigacji.",
                Toast.LENGTH_SHORT).show()
        }) {
            Icon(imageVector = Icons.Default.Menu, contentDescription = "Menu")
        }
    },
    colors = TopAppBarDefaults.smallTopAppBarColors(containerColor = Color.LightGray,
        titleContentColor = Color.Magenta)
)

Text("Hello World! Welcome to Tychy!", color = Color.White)

Text("Druga linia tekstu.", color = Color.White)

Divider(color = Color.White, thickness = 2.dp) // Linia, odpowiednik <hr>
// Wyrównanie elementów do prawej; inne wartości: Center
Row(horizontalArrangement = Arrangement.End,
    verticalAlignment = Alignment.CenterVertically,
    modifier = Modifier
        .fillMaxWidth()
        .padding(10.dp)
        .background(Color.Gray)) {
    Text("EL1", fontWeight = FontWeight.Bold, modifier = Modifier.padding(10.dp))
    Text("EL2", fontWeight = FontWeight.Bold, modifier = Modifier.padding(10.dp))
    Box(
        modifier = Modifier
            .size(100.dp)
            .padding(10.dp)
            .background(Color.Yellow),
        contentAlignment = Alignment.Center
    ) {
        Text("EL3")
    }
    // Uwaga: Pomiędzy elementami "Row" można także użyć "Spacer".
    Box(
        modifier = Modifier
            .size(100.dp)
            .padding(10.dp)
            .background(Color.Yellow),
        contentAlignment = Alignment.Center
    ) {
        Text("EL4")
    }
}

Spacer(modifier = Modifier.height(30.dp))
```

```
// Rozmiar tekstu poniżej może być podany w "em" lub (lepiej!) w "sp" (scale-independent
 pixels):
Text("Brajan Sheridan",
    color = Color.Red,
    fontSize = 30.sp,
    fontStyle = FontStyle.Italic,
    fontWeight = FontWeight.Bold,
    fontFamily = FontFamily.Cursive, // Inne wartości: Monospace, Default
    modifier = Modifier
        .background(Color.LightGray)
        .padding(all = 20.dp))
// Szerokość linii (długość linii):
// modifier = Modifier.width(150.dp)
// Maksymalna wysokość pola tekstowego:
// modifier = Modifier.height(50.dp)
// View.setBackgroundColor(Color.parseColor("#e7eecc"));

Spacer(modifier = Modifier.height(30.dp))
// Opcja "by remember" po to, aby zawartość zmiennej nie zmieniła się podczas zmiany
 stanu aplikacji.
// Zamiast "TextField" można użyć "OutlinedTextField".
var username by remember { mutableStateOf("") }
TextField(
    value = username,
    onValueChange = { username = it },
    label = { Text("Wpisz tekst:") }
)
Spacer(modifier = Modifier.height(30.dp))
Text(username)

Spacer(modifier = Modifier.height(30.dp))

Image(
    // Obrazy jpg / png należy umieścić w: app / src / main / res / drawable
    painter = painterResource(id = R.drawable.robot),
    contentDescription = null,
    // Skalowanie obrazu:
    contentScale = ContentScale.Fit, modifier = Modifier.size(100.dp)
    // Obramowanie obrazka:
    // modifier = Modifier.border(
    //        BorderStroke(2.dp, Color.Gray),
    //        RectangleShape
    //    )
    // Zaokrąglenie obrazu:
    // modifier = Modifier.clip(CircleShape)
    // Zaokrąglone narożniki obrazka:
    // modifier = Modifier.clip(RoundedCornerShape(30))
    // Przeźroczystość obrazka:
```

```
        // alpha = 0.2F
    )

    Image(painter = painterResource(id = R.drawable.kot),
    contentDescription = null,
    // Skalowanie obrazu:
    contentScale = ContentScale.Fit, modifier = Modifier.size(100.dp))

    Spacer(modifier = Modifier.height(30.dp))

    // Elevation to złudzenie, że element jest podniesiony lekko do góry (rzuca cień):
    Card(elevation = CardDefaults.cardElevation(defaultElevation = 4.dp),
        shape = RoundedCornerShape(20.dp),
        modifier = Modifier.padding(all = 10.dp),
        colors = CardDefaults.cardColors(containerColor = Color.Yellow)) {
        Column(modifier = Modifier.padding(all = 10.dp)) {
            Text("To jest karta.", fontWeight = FontWeight.W700)
            Text("Zawiera elementy w kolumnie,")
            Text("czyli: element pod elementem.", color = Color.Gray)
        }
    }

    Spacer(modifier = Modifier.height(30.dp))

    Button(
        // enabled = false,
        onClick = { Toast.makeText(contextForToast, "Przycisk został naciśnięty.",
            Toast.LENGTH_LONG).show() },
        modifier = Modifier.height(60.dp),
        border = BorderStroke(2.dp, Color.Gray),
        // contentPadding = PaddingValues(1.dp),
        shape = CircleShape,
        // Inne wartości: RectangleShape, CutCornerShape(percent = 25),
        // RoundedCornerShape(10.dp)
        // Elevation to symulowanie wciśniętego / zwolnionego przycisku:
        elevation = ButtonDefaults.buttonElevation(
            defaultElevation = 20.dp,
            pressedElevation = 5.dp,
            disabledElevation = 0.dp
        )
        // colors = ButtonDefaults.outlinedButtonColors(backgroundColor = Color.Yellow)
    ) {
        Text("Zatwierdź polecenie...", color = Color.Red)
    }
  }
}
```

```
// PODGLĄD:
@Preview(showBackground = true)
@Composable
fun Podglad() {
  NazwaProjektuTheme {
     MojeElementy()
  }
}
```

Ostatnia aktualizacja: 1 października 2023.