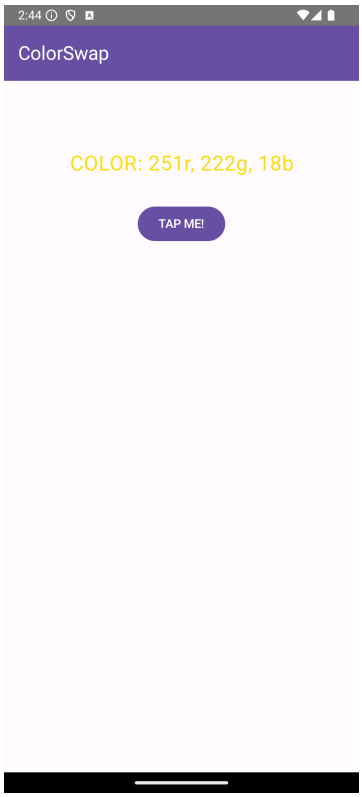
Homework 4

Lim Saeyeon (#21102054)

1. ColorSwap




```
class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            MaterialTheme {
                ColorSwapScreen()
            }
        }
    }
}
```

```
@OptIn(ExperimentalMaterial3Api::class)
@Composable
fun ColorSwapScreen() {
    var colorText by remember { mutableStateOf("Tap to Change Color") }
    var textColor by remember { mutableStateOf(Color.Black) }

    Scaffold(
        topBar = {
            TopAppBar(
                title = { Text("ColorSwap") },
                colors = TopAppBarDefaults.topAppBarColors(
                    containerColor = MaterialTheme.colorScheme.primary,
                    titleContentColor = Color.White
                )
            )
        }
    ) { paddingValues ->
        Column(
            modifier = Modifier
                .fillMaxSize()
                .padding(paddingValues)
                .padding(16.dp),
            horizontalAlignment = Alignment.CenterHorizontally,
            verticalArrangement = Arrangement.spacedBy(16.dp, Alignment.Top)
        ) {
            Spacer(modifier = Modifier.height(50.dp))

            Text(
                text = colorText,
                fontSize = 24.sp,
                color = textColor,
                textAlign = TextAlign.Center,
                modifier = Modifier.padding(bottom = 16.dp)
            )

            Button(
                onClick = {
                    val red = Random.nextInt(256)
                    val green = Random.nextInt(256)
                    val blue = Random.nextInt(256)

                    textColor = Color(red, green, blue)
                    colorText = "COLOR: ${red}r, ${green}g, ${blue}b"
                }
            ) {
                Text("TAP ME!")
            }
        }
    }
}
```

1) Main Components
- MainActivity : Entry point of the app, sets up the Compose UI
- ColorSwapScreen : Main composable function that contains the entire UI layout

2) UI Structure
- Top: Material3 TopAppBar with "ColorSwap" title
- Middle: Text showing current color values
- Bottom: "TAP ME!" button for color changes

3) State Management
- Uses `remember` and `mutableStateOf` for two states
  - colorText : Displays the RGB values
  - textColor : Controls the color of the text

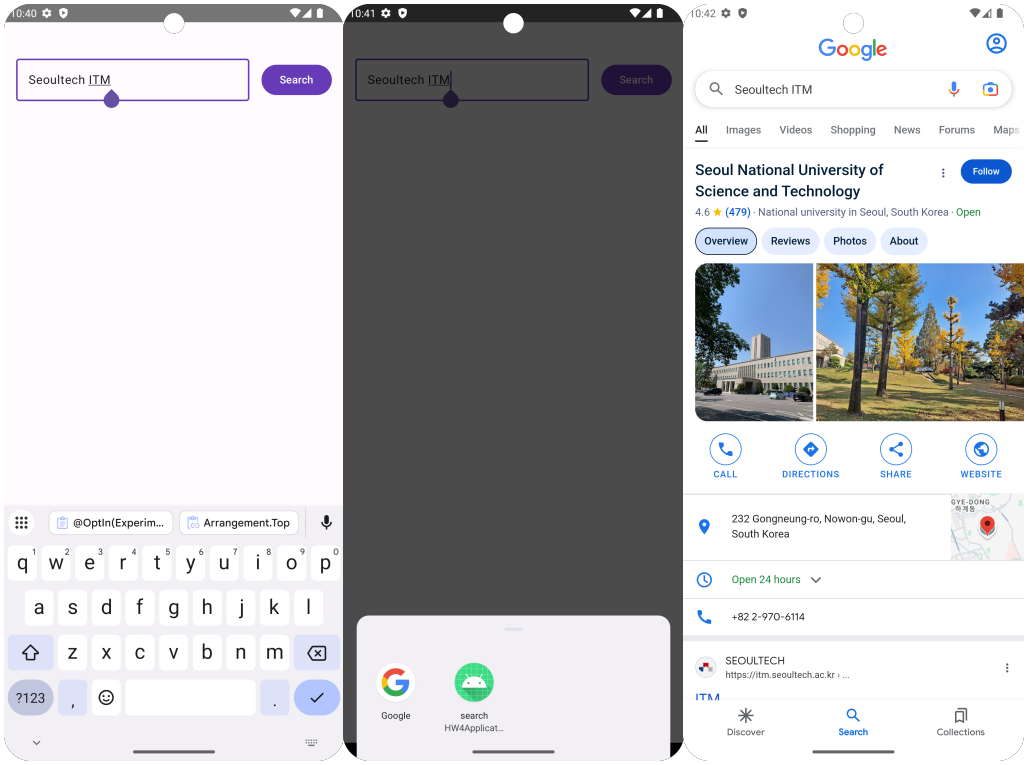
4) Layout
- Uses `Scaffold` for basic material design layout
- Column arrangement for vertical content alignment
- Centered horizontally using `Alignment.CenterHorizontally`

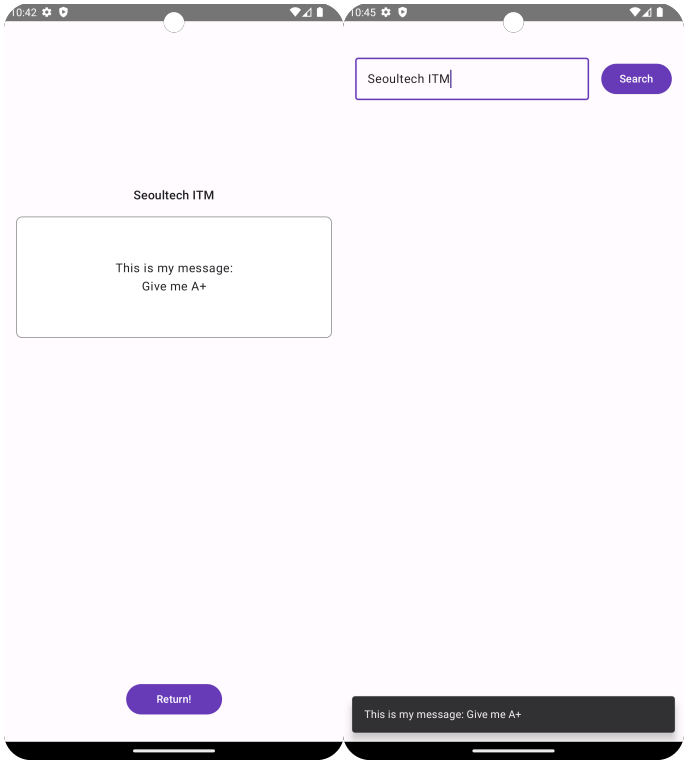
5.) Functionality
- Button click generates random RGB values (0-255)
- Updates both text content and color simultaneously
- Format : "COLOR: {red}r, {green}g, {blue}b"

6) Design Features
- Material3 components (TopAppBar, Button)
- Consistent padding and spacing
- Dynamic color changes
- White text on primary color app bar

* Problem 2 (Portfolio) is described in selfintro.zip

3. Search


Seoultech ITM
Search
Seoultech ITM
Search
Google
Seoultech ITM
All
Images
Videos
Shopping
News
Forums
Seoul National University of Science and Technology
Follow
4.6 (479) · National university in Seoul, South Korea · Open
Overview
Reviews
Photos
About
CALL
DIRECTIONS
SHARE
WEBSITE
232 Gongneung-ro, Nowon-gu, Seoul, South Korea
Open 24 hours
+82 2-970-6114
SEOULTECH
Discover
Search
Collections
Google
search HW4Applicat...


Seoultech ITM
Search
Seoultech ITM
This is my message:
Give me A+
Return!
This is my message: Give me A+

MainActivity.kt

```
class MainActivity : ComponentActivity() {
    private val getSearchResult = registerForActivityResult(
        ActivityResultContracts.StartActivityForResult()
    ) { result ->
        if (result.resultCode == Activity.RESULT_OK) {
            val message = result.data?.getStringExtra("user_message") ?: ""
            lifecycleScope.launch {
                snackbarHostState.showSnackbar(message)
            }
        }
    }

    private val snackbarHostState = SnackbarHostState()

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            MaterialTheme {
                MainScreen(getSearchResult, snackbarHostState)
            }
        }
    }
}


@Composable
fun MainScreen(
    getSearchResult: ActivityResultLauncher<Intent>,
    snackbarHostState: SnackbarHostState
) {
    var searchQuery by remember { mutableStateOf("") }

    Scaffold(
        snackbarHost = { SnackbarHost(snackbarHostState) }
    ) { padding ->
        Column(
            modifier = Modifier
                .fillMaxSize()
                .padding(padding)
                .padding(16.dp),
            horizontalAlignment = Alignment.CenterHorizontally,
            verticalArrangement = Arrangement.Top
        ) {
            Spacer(modifier = Modifier.height(32.dp))

            Row(
                modifier = Modifier.fillMaxWidth(),
                horizontalArrangement = Arrangement.spacedBy(16.dp),
                verticalAlignment = Alignment.CenterVertically
            ) {
                OutlinedTextField(
                    value = searchQuery,
                    onValueChange = { searchQuery = it },
                    modifier = Modifier.weight(1f),
                    singleLine = true,
                    colors = OutlinedTextFieldDefaults.colors(
                        unfocusedBorderColor = Color.Gray,
                        focusedBorderColor = Color(0xFF673AB7)
                    ),
                    shape = RoundedCornerShape(4.dp)
                )

                Button(
                    onClick = {
```

```
                        val searchIntent = Intent(Intent.ACTION_WEB_SEARCH).apply {
                            putExtra(SearchManager.QUERY, searchQuery)
                        }
                        val chooserIntent = Intent.createChooser(
                            searchIntent,
                            "Complete action using"
                        )
                        getSearchResult.launch(chooserIntent)
                    },
                    colors = ButtonDefaults.buttonColors(
                        containerColor = Color(0xFF673AB7)
                    ),
                    shape = RoundedCornerShape(24.dp)
                ) {
                    Text("Search")
                }
            }
        }
    }
}
```

1) Activity Structure
   - Uses Jetpack Compose for UI
   - Implements Activity Result API for handling search results
   - Uses SnackbarHostState for showing returned messages

2) Main Screen UI
   - Simple layout with search bar and button
   - Row arrangement for search components
   - Top-aligned layout with spacing

3) Key Components
   - OutlinedTextField for search input
   - Search button
   - Snackbar for displaying results
   - Weight-based row distribution

4) Functionality
   - Handles search input state
   - Creates web search intent
   - Shows app chooser dialog
   - Displays returned messages in snackbar

SubActivity.kt

```
class SubActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        val searchQuery = intent.getStringExtra(SearchManager.QUERY) ?: ""

        setContent {
            MaterialTheme {
                SubScreen(searchQuery) { userMessage ->
                    setResult(Activity.RESULT_OK, Intent().apply {
                        putExtra("user_message", userMessage)
                    })
                    finish()
                }
            }
        }
    }
}



@Composable
fun SubScreen(searchQuery: String, onReturn: (String) -> Unit) {
    Scaffold { padding ->
        Column(
            modifier = Modifier
                .fillMaxSize()
                .padding(padding),
            horizontalAlignment = Alignment.CenterHorizontally,
            verticalArrangement = Arrangement.SpaceBetween
        ) {
            Spacer(modifier = Modifier.weight(1f))

            Column(
                modifier = Modifier.weight(2f),
                horizontalAlignment = Alignment.CenterHorizontally
            ) {
                Text(
                    text = searchQuery,
                    style = MaterialTheme.typography.titleMedium,
                    modifier = Modifier.padding(bottom = 16.dp)
                )

                Surface(
                    modifier = Modifier
                        .fillMaxWidth()
                        .height(160.dp)
                        .padding(horizontal = 16.dp),
                    shape = RoundedCornerShape(8.dp),
                    border = BorderStroke(1.dp, Color.Gray),
                    color = Color.White
                ) {
                    Column(
                        modifier = Modifier
                            .fillMaxSize()
                            .padding(16.dp),
                        horizontalAlignment = Alignment.CenterHorizontally,
                        verticalArrangement = Arrangement.Center
                    ) {
                        Text(
                            text = "This is my message:",
                            style = MaterialTheme.typography.bodyLarge,
                            textAlign = TextAlign.Center
                        )
```

```
                        Text(
                            text = "Give me A+",
                            style = MaterialTheme.typography.bodyLarge,
                            textAlign = TextAlign.Center
                        )
                    }
                }
            }

            Spacer(modifier = Modifier.weight(1f))

            Button(
                onClick = { onReturn("This is my message: Give me A+") },
                colors = ButtonDefaults.buttonColors(
                    containerColor = Color(0xFF673AB7)
                ),
                shape = RoundedCornerShape(24.dp),
                modifier = Modifier.padding(bottom = 32.dp)
            ) {
                Text(
                    "Return!",
                    modifier = Modifier.padding(horizontal = 16.dp)
                )
            }
        }
    }
}
```

Here's a short analysis of the SubActivity code:

1) Layout Structure
- Uses Compose's Scaffold for basic layout
- Three-part vertical layout using weight distribution
- Centered alignment for all components

2) Key Components
- Search query display at top
- Message box in center with fixed height
- Return button at bottom
- Spacers for flexible spacing

3) Functionality
- Receives search query as parameter
- Displays fixed message "This is my message: Give me A+"
- Returns combined message via callback
- Uses Activity result pattern for communication

AndroidManifest.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.Search"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true"
            android:label="@string/app_name"
            android:theme="@style/Theme.Search">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".SubActivity"
            android:exported="true"
            android:label="HW4Application">
            <intent-filter>
                <action android:name="android.intent.action.WEB_SEARCH" />
                <action android:name="android.intent.action.SEARCH" />
                <category android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

1) Main Components
   - Two activities declared: MainActivity and SubActivity
   - MainActivity is the launcher activity (entry point)
   - SubActivity handles web search intents

2) MainActivity Configuration
   - Exported (can be launched by other apps)
   - Set as main launcher via intent-filter
   - Uses "Search" theme

3) SubActivity Configuration
   - Labeled as "HW4 Application"
   - Handles two intent types:
     - android.intent.action.WEB_SEARCH
     - android.intent.action.SEARCH
   - Includes DEFAULT category to receive implicit intents
   - Exported for external access

Logic Description of Search App

The search app serves as a demonstration of Android's inter-app communication capabilities and activity result handling mechanisms. The application is built through Jetpack Compose, and showcasing proper state management.

When launched, the app presents a minimalist interface through MainActivity, featuring a search bar and a purple search button. This clean design follows Material Design guidelines, with the search functionality centrally positioned at the top of the screen. The user interface is responsive, providing immediate feedback through visual cues like the purple accent color when the search field is focused.

Upon entering a search query and pressing the "Search" button, the app creates an implicit intent that triggers Android's system chooser dialog. This dialog presents two options to the user: the Chrome browser for performing web searches, or the app's own custom handler (SubActivity) labeled as "HW4".

If Chrome is selected, the search query is passed to the browser, which performs a standard web search. The user can then browse the results and return to the app using the system back button. However, if the HW4 option is chosen, the search query is passed to SubActivity via intent extras, demonstrating inter-activity communication within the app.

The SubActivity presents a three-part layout : the original search query at the top, a bordered box containing the fixed message "This is my message: Give me A+" in the center, and a purple "Return!" button at the bottom. When the user presses the return button, the message is sent back to MainActivity using the modern Activity Result API. Then MainActivity receives the returned message and displays it in a Material Design Snackbar at the bottom of the screen.