Name: Rahul Kumar S

RegNo: 20BCE1878

Code:

MainActivity.kt

```
package com.plcoding.videoplayercompose

import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.rememberLauncherForActivityResult
import androidx.activity.compose.setContent
import androidx.activity.result.contract.ActivityResultContracts
import androidx.compose.foundation.clickable
import androidx.compose.foundation.layout.*
import androidx.compose.foundation.lazy.LazyColumn
import androidx.compose.foundation.lazy.items
import androidx.compose.material.*
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.FileOpen
import androidx.compose.runtime.*
import androidx.compose.ui.Modifier
import androidx.compose.ui.platform.LocalLifecycleOwner
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.dp
import androidx.compose.ui.viewinterop.AndroidView
import androidx.hilt.navigation.compose.hiltViewModel
import androidx.lifecycle.Lifecycle
import androidx.lifecycle.LifecycleEventObserver
import androidx.lifecycle.LifecycleOwner
import androidx.media3.ui.PlayerView
import com.plcoding.videoplayercompose.ui.theme.VideoPlayerComposeTheme
import dagger.hilt.android.AndroidEntryPoint

@AndroidEntryPoint
class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            VideoPlayerComposeTheme {
                val viewModel = hiltViewModel<MainViewModel>()
                val videoItems by viewModel.videoItems.collectAsState()
                val selectVideoLauncher = rememberLauncherForActivityResult(
                    contract = ActivityResultContracts.GetContent(),
                    onResult = { uri ->
                        uri?.let(viewModel::addVideoUri)
                    }
                )
                var lifecycle by remember {
                    mutableStateOf(Lifecycle.Event.ON_CREATE)
                }
                val lifecycleOwner = LocalLifecycleOwner.current
                DisposableEffect(lifecycleOwner) {
                    val observer = LifecycleEventObserver { _, event ->
                        lifecycle = event
                    }
                    lifecycleOwner.lifecycle.addObserver(observer)
```

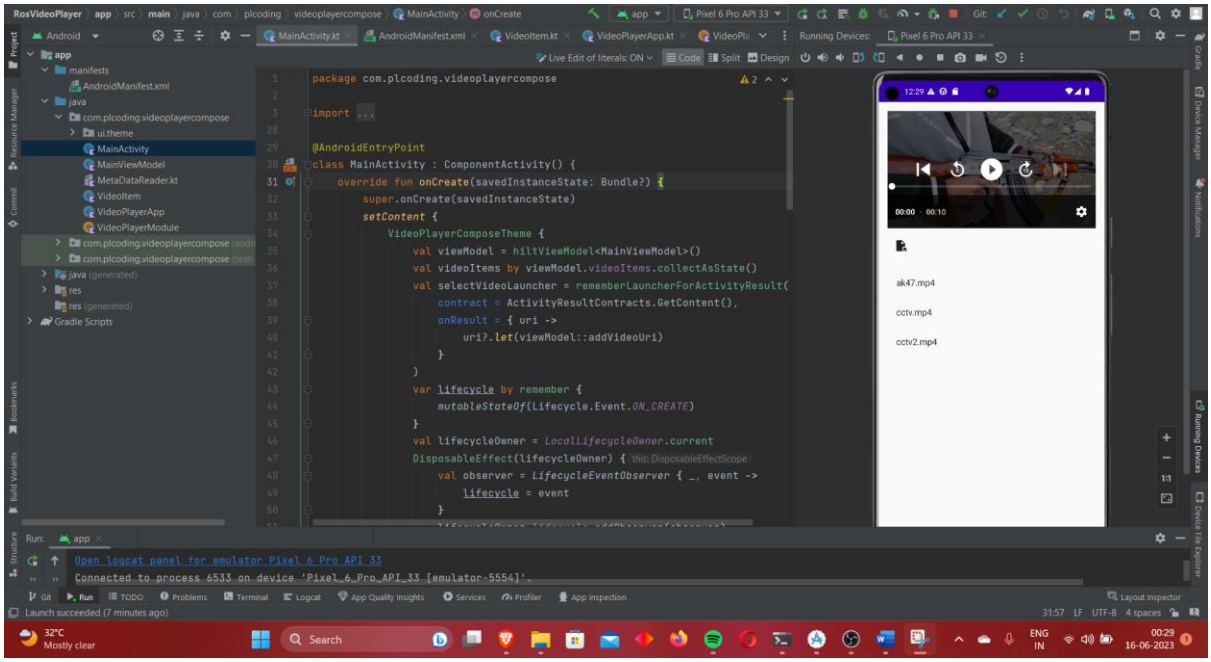
```
            onDispose {
                lifecycleOwner.lifecycle.removeObserver(observer)
            }
        }

        Column(
            modifier = Modifier
                .fillMaxSize()
                .padding(16.dp)
        ) {
            AndroidView(
                factory = { context ->
                    PlayerView(context).also {
                        it.player = viewModel.player
                    }
                },
                update = {
                    when (lifecycle) {
                        Lifecycle.Event.ON_PAUSE -> {
                            it.onPause()
                            it.player?.pause()
                        }
                        Lifecycle.Event.ON_RESUME -> {
                            it.onResume()
                        }
                        else -> Unit
                    }
                },
                modifier = Modifier
                    .fillMaxWidth()
                    .aspectRatio(16 / 9f)
            )
            Spacer(modifier = Modifier.height(8.dp))
            IconButton(onClick = {
                selectVideoLauncher.launch("video/mp4")
            }) {
                Icon(
                    imageVector = Icons.Default.FileOpen,
                    contentDescription = "Select video"
                )
            }
            Spacer(modifier = Modifier.height(16.dp))
            LazyColumn(
                modifier = Modifier.fillMaxWidth()
            ) {
                items(videoItems) { item ->
                    Text(
                        text = item.name,
                        modifier = Modifier
                            .fillMaxWidth()
                            .clickable {
                                viewModel.playVideo(item.contentUri)
                            }
                            .padding(16.dp)
                    )
                }
            }
        }
    }
```

```
        }
    }
}
```

MainViewModel.kt

```
package com.plcoding.videoplayercompose

import android.net.Uri
import androidx.lifecycle.SavedStateHandle
import androidx.lifecycle.ViewModel
import androidx.lifecycle.viewModelScope
import androidx.media3.common.MediaItem
import androidx.media3.common.Player
import dagger.hilt.android.lifecycle.HiltViewModel
import kotlinx.coroutines.flow.SharingStarted
import kotlinx.coroutines.flow.map
import kotlinx.coroutines.flow.stateIn
import javax.inject.Inject

@HiltViewModel
class MainViewModel @Inject constructor(
    private val savedStateHandle: SavedStateHandle,
    val player: Player,
    private val metaDataReader: MetaDataReader
): ViewModel() {

    private val videoUris = savedStateHandle.getStateFlow("videoUris", emptyList<Uri>())

    val videoItems = videoUris.map { uris ->
        uris.map { uri ->
            VideoItem(
                contentUri = uri,
                mediaItem = MediaItem.fromUri(uri),
                name = metaDataReader.getMetaDataFromUri(uri)?.fileName ?: "No name"
            )
        }
    }.stateIn(viewModelScope, SharingStarted.WhileSubscribed(5000), emptyList())

    init {
        player.prepare()
    }

    fun addVideoUri(uri: Uri) {
        savedStateHandle["videoUris"] = videoUris.value + uri
        player.addMediaItem(MediaItem.fromUri(uri))
    }

    fun playVideo(uri: Uri) {
        player.setMediaItem(
            videoItems.value.find { it.contentUri == uri }?.mediaItem ?: return
        )
    }

    override fun onCleared() {
        super.onCleared()
        player.release()
    }
}
```

Output


package com.plcoding.videoplayercompose
import ...
@AndroidEntryPoint
class MainActivity : ComponentActivity() {
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
setContent {
VideoPlayerComposeTheme {
val viewModel = hiltViewModel<MainViewModel>()
val videoItems by viewModel.videoItems.collectAsState()
val selectVideoLauncher = rememberLauncherForActivityResult(
contract = ActivityResultContracts.GetContent(),
onResult = { uri ->
uri?.let(viewModel::addVideoUri)
}
)
var lifecycle by remember {
mutableStateOf(Lifecycle.Event.ON_CREATE)
}
val lifecycleOwner = LocalLifecycleOwner.current
DisposableEffect(lifecycleOwner) {
val observer = LifecycleEventObserver { _, event ->
lifecycle = event
}
ak47.mp4
cctv.mp4
cctv2.mp4
Open logcat panel for emulator Pixel 6 Pro API 33
Connected to process 6533 on device 'Pixel_6_Pro_API_33 [emulator-5554]'.
Launch succeeded (7 minutes ago)
32°C
Mostly clear
Search
ENG
IN
00:29
16-06-2023