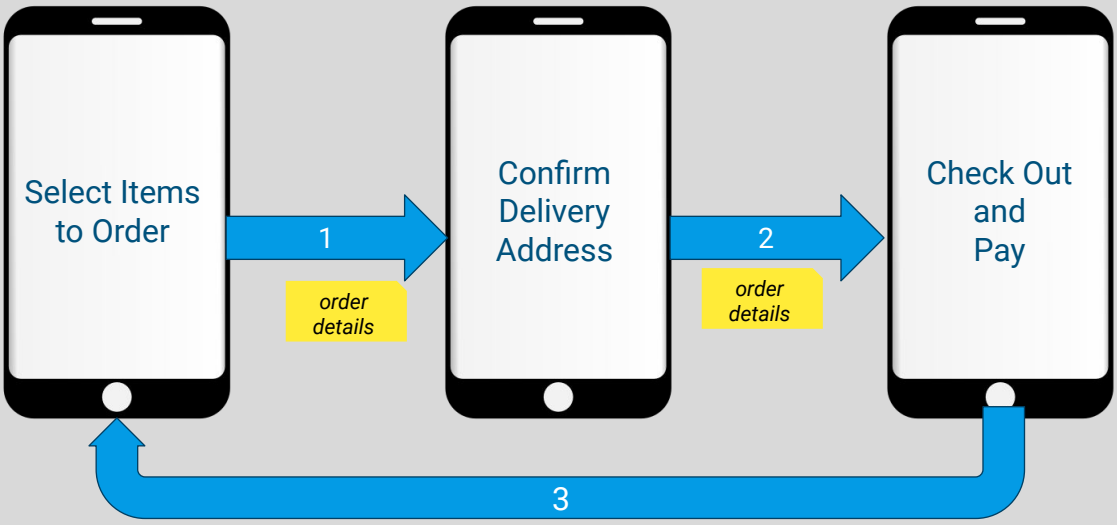
Navigation Basics in Jetpack Compose

Navigate from Source to Destination

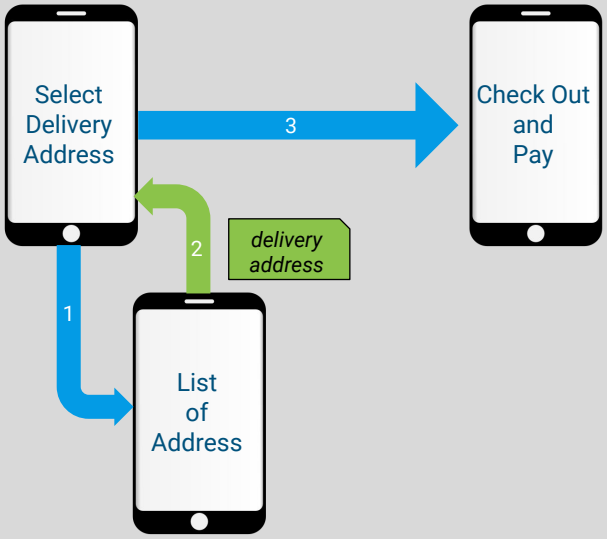
nav·i·gate

1. plan and direct the route of a ship, aircraft, or other form of transportation, especially by using instruments or maps
2. travel on a desired course after planning a route
3. guide over a specific route or terrain
4. *move from one accessible page, section, or view of a file or website to another*

Use Case #1: Navigate Forward with Data



Use Case #2: Navigate Forward, Return with Data



Activity Navigation + Compose

- Navigation Between Composable within a Single Activity
  - Passing data payload forward (from source to destination)
  - Returning data payload backward (from destination to source)
- Navigation Across Activities
  - The UI of each activity is built using Jetpack Compose
  - Passing data payload forward
  - Returning data payload backward

UI Navigation in Jetpack Compose

Elements of Navigation

Road Trip	UI Navigation
City of origin	@Composable Screen of origin
City of destination	@Composable Screen of destination
Map	Navigation Host
Roads	Routes
Driver	Navigation Controller

Compose Navigation Router

- Required Components
  - Source/Destination: UI @Composable functions
  - A Navigation Host (@Composable) at the “top-level” Composable
  - An instance of a NavigationController
  - Define road to reach destination: entry point of each destination as a tuple: (*routename*, composableName)
- Similar to Routers in Web Apps
  - React Router
  - Vue Router

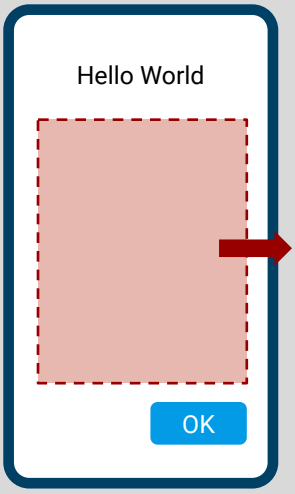
Step 1: Prepare Your Trips

```
val navController = rememberNavController() // Decide who drives!!

NavHost (navController, startDestination = “_____”) {
    composable(“tripToDetroit”) {
        DetroitTour(navController)
    }
    composable(“chicagoTour”) {
        ChicagoSkyline(navController)
    }
    composable(“lasVegas/{cashAmount}/{creditCard}”) {
        var myCash = 0
        it.arguments?.let {
            myCash = it.getInt(“cashAmount”)
        }
        GambleInVegas(navController, myCash)
    }
}
```

```
@Composable
fun DetroitTour(nc: NavHostController)
{
  // More code here
}
```

Navigation Host: “road space” for your trips

```
Column {
    Text(“Hello World”)
    val nc = rememberNavController()
    NavHost(navController = nc) {
      composable(“one”) {
         FirstDestination()
      }
      composable(“two”) {
         SecondDestination()
      }
    }
    Button {  Text(“OK”) }
}

@Composable
fun FirstDestination() {
}
```



This screen area will be used for rendering the UI of FirstDestination() or SecondDestination()

Step 2: Navigate To (a specific) Destination

```
NavHost (navController, startDestination = “_____”) {
    composable(“tripToDetroit”) {
        DetroitTour(navController)
    }
    composable(“lasVegas/{cashAmount}/{creditCard}”) {
        var myCash = 0
        it.arguments?.let {
            myCash = it.getInt(“cashAmount”)
        }
        GambleInVegas(navController, myCash)
    }
}
```

```
Button(onClick = {
  navCtrl.navigate(“tripToDetroit”)
  }
) {
   Text(“DTW”)
}
```

```
Button(onClick = {
  navCtrl.navigate(“lasVegas/3000/1234 5678 xxxx yyyy”)
  }
) {
   Text(“Vegas”)
}
```

Details of NavHost & Composables

```
NavHost(navController = nc,
        startDestination = “one”) {
  composable(“one”) {
    FirstDestination(____, navCtrl = nc)
  }
  composable(“two”) {
    SecondDestination(____, navCtrl = nc)
  }
}
```

```
@Composable
fun FirstDestination(/* more params */,
    navCtrl: NavController)
{
   Button(onClick = {
     navCtrl.navigate(“two”)
   }) {
     Text(“Go”)
   }
}

@Composable
fun SecondDestination(/* more params */,
    navCtrl: NavController)
{
   navCtrl.popBackStack()
}
```

Step 3: Return Home from Road Trip

GitHub
android-compose-navigation

Sending Data (Forward) via Route

From MainScreen to TicketPurchase

```
NavHost(navController = nc, startDestination = “one”) {
  // Route #1
  composable(“one”) {
    MainScreen(____, navCtrl = nc)
  }

  // Route #2
  composable(“tickets/{numTix}/{creditCard}”) {
    // Unpack the args
    val tix = it.arguments?.getString(“numTix”)
    val cc = it.arguments?.getString(“creditCard”)

    TicketPurchase(tix.toInt() ?: 0,
                   creditNumber ?: “none”,
                   navCtrl = nc)
  }
}
```

```
@Composable
fun MainScreen(___, navCtrl: NavController) {
  Button(onClick = {
    navCtrl.navigate(“tickets/10/12345678______”)
  }) {
     Text(“Confirm Purchase”)
  }
}

@Composable
fun TicketPurchase (count: Int, payment: String,
  navCtrl: NavController) {
}
```

Sending Data Backward

From TicketPurchase to MainScreen

```
NavHost(navController = nc, startDestination = “one”) {
  composable(“one”) {
    val tx = it.savedStateHandle.get<String>(“tixConfirmation”)
    it.savedStateHandler.remove<String>(“tixConfirmation”)
    MainScreen(____, navCtrl = nc)
  }

  // Route #2
  composable(“two”) {
    TicketPurchase(_____, navCtrl = nc)
  }
}
```

```
@Composable
fun TicketPurchase (count: Int, payment: String,
  navCtrl: NavController) {
  /* inside a button click handler */
  Button(onClick = {
    navCtrl.previousBackStackEntry
           ?.saveStateHandle
           ?.set(“tixConfirmation”, “GTW451-99”)
    navCtrl.popBackStack()
  }) {
    Text(“Confirm”)
  }
}
```

Type Safe Navigation

```
implementation("androidx.navigation:navigation-compose:2.8.1")
implementation("androidx.navigation:navigation-runtime-ktx:2.8.1")
```

Navigation with Intent

Navigation: Sending Data Payload Forward

```
// Without Compose
okButton.setOnClickListener {
    val toNext = Intent(this, NextActivity::class.java)
    toNext.putExtra ("atom", "Oxygen")
    toNext.putExtra ("weight", 15.99f)
    startActivity(toNext)
}
```

```
// With Jetpack Compose, inside a @Composable function
val thisContext = LocalContext.current
val thisActivity = thisContext as? Activity
Button(onClick = {
      val toNext = Intent(thisContext, NextActivity::class.java)
      toNext.putExtra ("atom", "Oxygen")
      toNext.putExtra ("weight", 15.99f)
      thisActivity?.startActivity(toNext)
  }) {
  Text ("OK")
}
```

Navigation: Sending Result (Backward)

```
// Destination Activity (without Compose)
class DestActivity: AppCompatActivity() {

  fun somButtonHandler() {
    // Prepare result and send it back
    val pack = Intent()
    pack.putExtra("spins", "_____")
    setResult(RESULT_OK, pack)
    finish()
  }
}
```

```
// Destination Activity (with Compose)
class DestinationActivity: ComponentActivity() {
  override fun onCreate(___) {
    super.onCreate(____)
    setContent {
      MyDestUI()
    }
  }
}

@Composable
fun MyComposableUI() {
  val thisContext = LocalContext.current
  val thisActivity = thisConteext as? Activity
  Button(onClick = {
    // Prepare result and send it back
    val pack = Intent()
    pack.putExtra("spins", "_____")
    thisActivity?.setResult(RESULT_OK, pack)
    thisActivity?.finish()

  }) {
    Text("OK")
  }
}
```

Navigation: Return to Previous Activity

```
// Without Compose
doneButton.setOnClickListener {
    finish()
}
```

```
// With Jetpack Compose, inside a @Composable function
val thisContext = LocalContext.current
val thisActivity = thisContext as? Activity
Button(onClick = {
    thisActivity.finish()
  }) {
    Text ("Done")
}
```

Navigation: Receiving Data Payload

```
// Without Compose
class NextActivity: AppCompatActivity() {
  val vm: MyViewModel by viewmodels
  private var theAtom: String = ""
  private var theWeight: Float = 0.0f
  override fun onCreate(z: Bundle) {
    super.onCreate(z)
    // Use theAtom and theWeight
    // save theAtom and theWeight to a viewModel
  }

  override fun onResume() {
    super.onResume()
    theAtom = intent?.getStringExtra("atom")
    theWeight = intent?.getFloatExtra("weight", 0.0f)
  }
}
```

```
// With Compose
class NextActivity: ComponentActivity() {
  override fun onCreate(z: Bundle) {
    super.onCreate(z)
    setContent {
      val theAtom = intent?.getStringExtra("atom")
      val theWeight = intent?.getFloatExtra("weight", 0f):
      MyComposableUI(theAtom, theWeight)
  }
}

@Composable
fun MyComposableUI(a: String?, w: Float?) {
  // UI details go here
}
```

Navigation: Preparing for Backward Result

```
// Originating Activity (without Compose)
class OriginActivity: AppCompatActivity() {
  val nextLauncher = registerForActivityResult(
    ActivityResultContracts.StartActivityForResult()) {
      if (it.resultCode == Activity.RESULT_OK) {
        val ___ = it.data?.getStringExtra(“spins”)
      }
    }

  fun somButtonHandler() {
    // Transition to NextActivity & send data
    val toNext = Intent(this, NextActivity::class.java)
    nextLauncher.launch(toNext)
  }
}
```

```
// Originating Activity (with Compose)
class OriginActivity: ComponentActivity() {
  override fun onCreate(___) {
    super.onCreate(____)
    setContent {
      MyComposableUI()
    }
  }
}

@Composable
fun MyComposableUI() {
  val nextLauncher = remberLauncherForForActivityResult(
    ActivityResultContracts.StartActivityForResult()) {
      if (it.resultCode == Activity.RESULT_OK) {
        val ___ = it.data?.getStringExtra(“spins”)
      }
    }
  Button(onClick = {
    // Transition to NextActivity & send data
    val toNext = Intent(this, NextActivity::class.java)
    nextLauncher.launch(toNext)
  }) {
    Text(“OK”)
  }
}
```