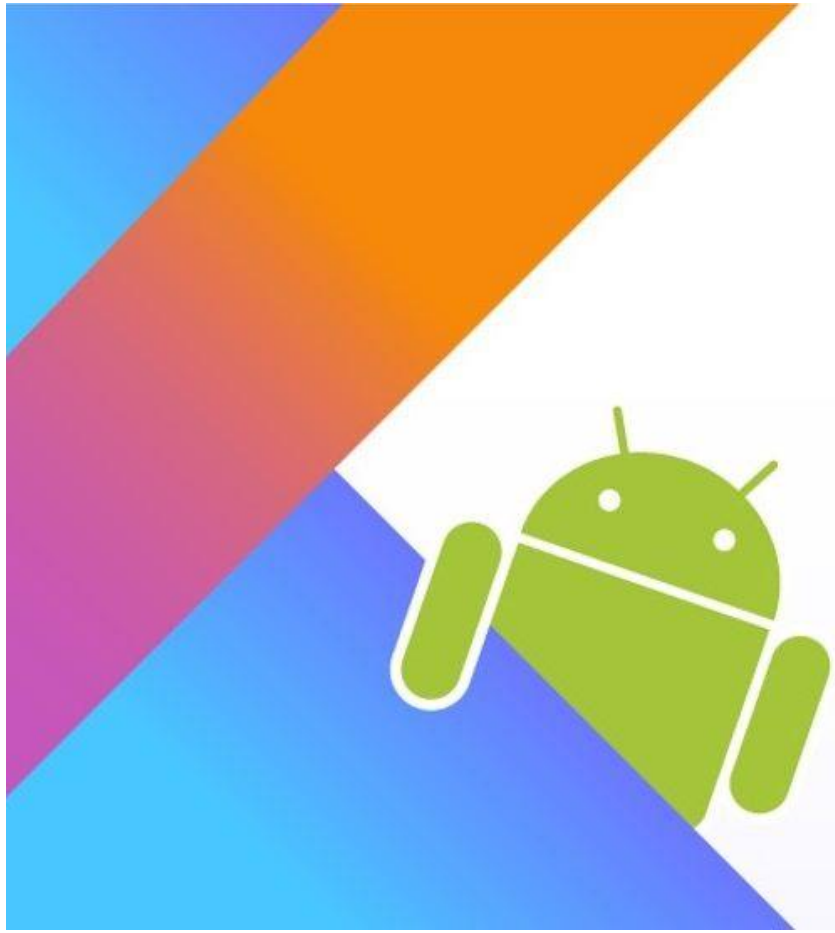
UNI GUAIRACÁ
CENTRO UNIVERSITÁRIO



Kotlin
for Android App Development

DESENVOLVIMENTO MOBILE

ANDROID – PASSANDO ARGUMENTOS

```
fun detalhes(v: View) {
    val dado = dados[adapterPosition]

    val args = Bundle()
    args.putString("titulo", dado.titulo)
    args.putString("descricao", dado.descricao)

    v.findNavController().navigate(R.id.lista_para_detalhes, args)
}
```

ANDROID – PASSANDO ARGUMENTOS

```
val titulo = arguments?.get("titulo")
val descricao = arguments?.get("descricao")

titulo_afazer.setText(titulo.toString())
descricao_afazer.setText(descricao.toString())
```

ANDROID – PASSANDO ARGUMENTOS

```
val action = SpecifyAmountFragmentDirections.confirmationAction(amount)
v.findNavController().navigate(action)
```

```
val args: ConfirmationFragmentArgs by navArgs()

override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
    val tv: TextView = view.findViewById(R.id.textViewAmount)
    val amount = args.amount
    tv.text = amount.toString()
}
```

ANDROID – ROOM

ORM (Object-Relational Mapping) oficial do Android

Utiliza como banco de dados SQL o SQLITE

Várias conveniências para salvar e carregar dados

ANDROID – ROOM

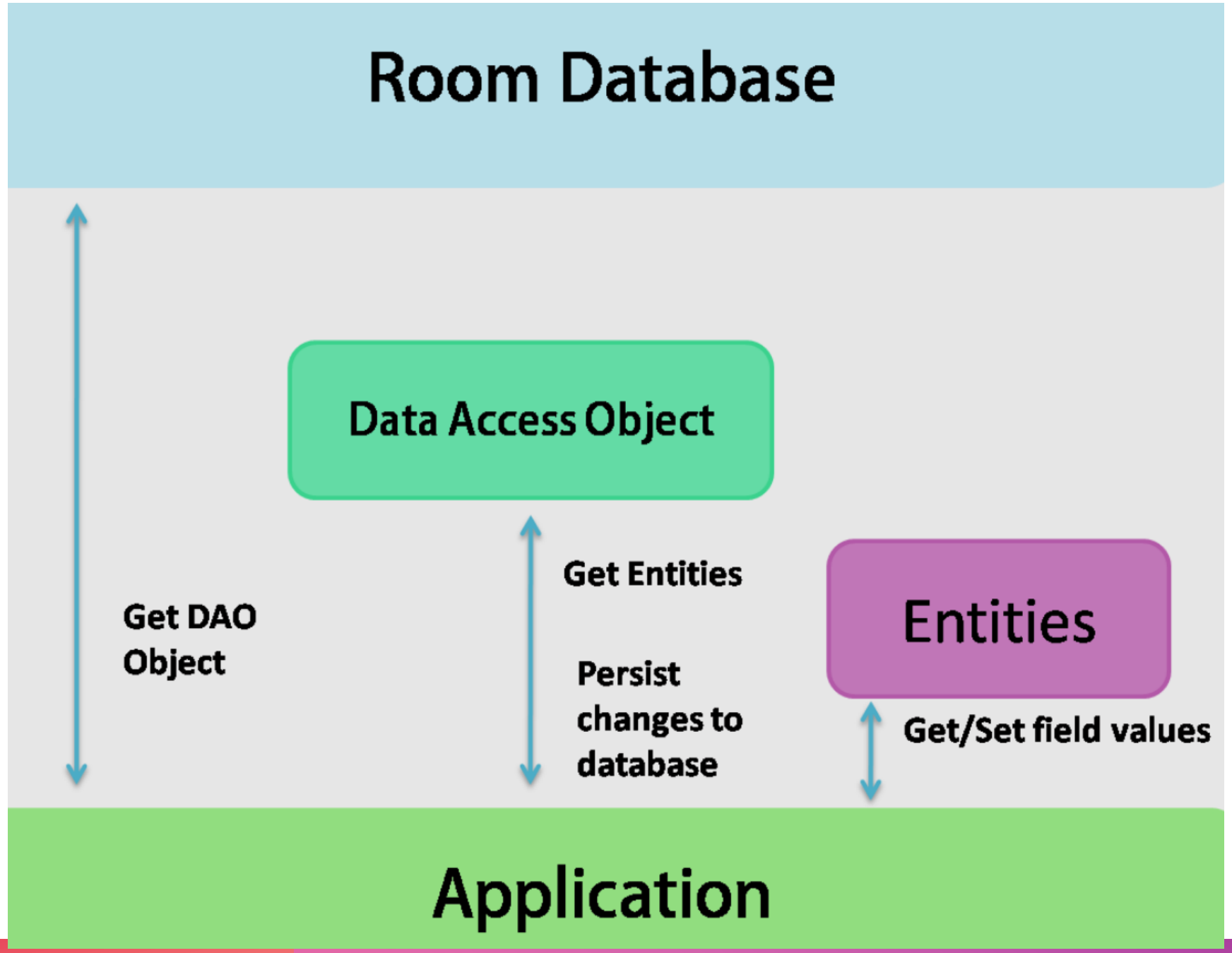
Room é feito pelo Trio de Componentes:

- Entities: Representa uma tabela (e 1 dado dela)

- Data Access Object (DAO): Guarda as operações de tabela

- Database: Constrói a tabela e da acesso a ela

ANDRO


Room Database
Data Access Object
Get DAO Object
Get Entities
Persist changes to database
Entities
Get/Set field values
Application

ANDROID – ROOM

Dependências no gradle(:app) :

```
apply plugin: 'kotlin-kapt'

dependencies {

    ....

    def room_version = "2.2.6"
    implementation "androidx.room:room-runtime:$room_version"
    kapt "androidx.room:room-compiler:$room_version"
    // optional - Kotlin Extensions and Coroutines support for Room
    implementation "androidx.room:room-ktx:$room_version"
}
```

ANDROID – ROOM

Entities: Representa uma tabela (e 1 dado dela)

```
import androidx.room.PrimaryKey

@Entity//(tableName = "meusAfazeres")
data class Afazer(
        //@ColumnInfo(name = "o_titulo")
        val titulo: String,
        val descricao: String) {

    @PrimaryKey(autoGenerate = true)
    var id: Long = 0
```

ANDROID – ROOM

Data Access Object: Guarda as operações de tabela

ANDROID – ROOM

```
@Dao
interface AfazerDao {
    @Query( value: "SELECT * FROM afazer WHERE id = :id ORDER BY id DESC")
    fun get(id: Long): Afazer

    @Query( value: "SELECT * FROM afazer")
    fun getAll(): List<Afazer>

    @Insert
    fun insert(afazer: Afazer): Long

    @Delete
    fun delete(afazer: Afazer)
}
```

ANDROID – ROOM

Database: Constrói a tabela e da acesso a ela

```
@Database(entities = arrayOf(Afazer::class), version = 1)
abstract class AppDatabase: RoomDatabase() {
    abstract fun afazerDao(): AfazerDao
}
```

ANDROID – ROOM

Database: Constrói a tabela e da acesso a ela

```
@Database(entities = arrayOf(Afazer::class), version = 1)
abstract class AppDatabase: RoomDatabase() {
    abstract fun afazerDao(): AfazerDao
}
```

ANDROID – ROOM

Database: Obtendo para usar

ANDROID – ROOM

Data

```
lateinit var db: AppDatabase

override fun onAttach(context: Context) {
    super.onAttach(context)

    db = Room.databaseBuilder(
        context,
        AppDatabase::class.java, name: "listadb"
    ).allowMainThreadQueries().build()
}
```

ANDROID – ROOM

Data

```
lateinit var db: AppDatabase

override fun onAttach(context: Context) {
    super.onAttach(context)

    db = Room.databaseBuilder(
        context,
        AppDatabase::class.java, name: "listadb"
    ).allowMainThreadQueries().build()
}
```

```
val adaptador = ListaAdapter(db.afazerDao())
```

ANDROID – ROOM

Data

```
lateinit var db: AppDatabase

override fun onAttach(context: Context) {
    super.onAttach(context)

    db = Room.databaseBuilder(
        context,
        AppDatabase::class.java, name: "listadb"
    ).allowMainThreadQueries().build()
}
```

```
val adaptador = ListaAdapter(db.afazerDao())
```

ANDROID – ROOM

Database: Obtendo para usar

```
class ListaAdapter(dao: AfazerDao)
    : RecyclerView.Adapter<ListaAdapter.AFazerHolder>() {

    val dados: MutableList<Afazer>
    init {
        dados = dao.getAll().toMutableList()
    }
```

ANDROID – ROOM

Usando no Adaptador:

A

```
fun apagar(position: Int) {
    val afazer = dados[position]
    dao.delete(afazer)
    dados.removeAt(position)
    notifyItemRemoved(position)
    notifyItemRangeChanged(position, itemCount: itemCount - position)
}

fun adicionar(titulo: String) {
    val afazer = Afazer(titulo, descricao: "Descrição...")
    val id = dao.insert(afazer)
    afazer.id = id
    dados.add( index: 0, afazer)
    notifyItemInserted( position: 0)
    notifyItemRangeChanged( positionStart: 0, itemCount)
}
```