



Observable Data Holders

**StateFlow, SharedFlow or
LiveData?**

We have a lot of options.









Great
question
pupils.
I prepared
some examples
for you.





*First, let's
compare them:*

Feature	State Flow	Live Data
Requires initial value in the constructor		
Automatically unregister the consumer in onStop		
Active instance exists independently of the presence of collectors (HOT)		

Declaration:

```
class MainViewModel : ViewModel() {  
  
    private val _placesAlreadyVisitedLiveData = MutableLiveData<List<String>>()  
    val placeAlreadyVisitedLiveData: LiveData<List<String>> = _placesAlreadyVisitedLiveData  
  
    private val _placesAlreadyVisitedStateFlow = MutableStateFlow<List<String>>(emptyList())  
    val placesAlreadyVisitedStateFlow: StateFlow<List<String>> = _placesAlreadyVisitedStateFlow
```

```
// more code ...
```

Producing events:

```
// live data  
_placesAlreadyVisitedLiveData.postValue(_placesVisited)  
  
// state flow  
viewModelScope.launch { this: CoroutineScope  
    _placesAlreadyVisitedStateFlow.emit(_placesVisited)  
}
```

Consuming
events:

```
class MainActivity : AppCompatActivity() {  
  
    lateinit var viewModel: MainViewModel  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)  
  
        viewModel = ViewModelProvider( owner: this)[MainViewModel::class.java]  
  
        LiveData  
        viewModel.placeAlreadyVisitedLiveData.observe( owner: this) { it: List<String>!  
            Log.d(TAG, msg: "onCreate: LiveData: $it ")  
        }  
  
        StateFlow  
        lifecycleScope.launch { this: CoroutineScope  
            viewModel.placesAlreadyVisitedStateFlow.collect { it: List<String>  
                Log.d(TAG, msg: "onCreate: StateFlow: $it ")  
            }  
        }  
    }  
}
```


*Dungeon Master, and about
SharedFlow?*



StateFlow has an operator to convert to a ***SharedFlow***:

```
class MainActivity : AppCompatActivity() {
```

```
// more code ...
```

```
    Handler(Looper.getMainLooper()).postDelayed({
        lifecycleScope.launch { this: CoroutineScope
            viewModel.placesAlreadyVisitedStateFlow.shareIn(
                lifecycleScope,
                started = SharingStarted.WhileSubscribed()
            ).collect { it: List<String>
                Log.d(TAG, msg: "onCreate: After 8s, SharedFlow: $it ")
            }
        }
    }, delayMillis: 8000L)
```

*Remember, **SharedFlow** is **COLD**, so if we compared to a **Live Data** we have something like that:*

```
class MainActivity : AppCompatActivity() {
```

```
// more code ...
```

```
Handler(Looper.getMainLooper()).postDelayed({
    lifecycleScope.launch { this: CoroutineScope
        viewModel.placesAlreadyVisitedStateFlow.shareIn(
            lifecycleScope,
            started = SharingStarted.WhileSubscribed()
        ).collect { it: List<String>
            Log.d(TAG, msg: "onCreate: After 8s, SharedFlow: $it ")
        }
    }
}, delayMillis: 8000L)
```

MainActivity	D onCreate: LiveData: [Canada]
MainActivity	D onCreate: LiveData: [Canada, USA]
MainActivity	D onCreate: LiveData: [Canada, USA, Panamá]
MainActivity	D onCreate: LiveData: [Canada, USA, Panamá, Mexico]
MainActivity	D onCreate: LiveData: [Canada, USA, Panamá, Mexico, Colombia]
MainActivity	D onCreate: LiveData: [Canada, USA, Panamá, Mexico, Colombia, Argentina]
MainActivity	D onCreate: LiveData: [Canada, USA, Panamá, Mexico, Colombia, Argentina, Brazil]
MainActivity	D onCreate: LiveData: [Canada, USA, Panamá, Mexico, Colombia, Argentina, Brazil, Italy]
MainActivity	D onCreate: After 8s, SharedFlow: [Canada, USA, Panamá, Mexico, Colombia, Argentina, Brazil, Italy]



*Dungeon Master, how to test
SharedFlow, **StateFlow**,
LiveData?*

*DUNGEON MASTER, where are
you?*

I hate when he does that.



By Moro