Android : Coroutines flow news ticker

코루틴 플로우, 레트로핏, MVVM

- 코루틴을 사용해서 UI 를 업데이트 하자
- 레트로핏과 플로우를 같이 사용해보자.
- MVVM 에서 어떻게 적용되는가
- **뉴스 아이템을 periodic 하게 리스트에 보여준다.**

0. 그래이들 디펜던시 추가, xml 셋팅

1. Retrofit 사용 준비하기
    1. NewsArticle 모델 클래스 작성
    2. NewsService 클래스로 Retrofit 객체 만든 후 Flow 로 객체 emit 받기
    3. NewsRepository 에서 getNews 메서드 호출해서 Flow 시작시키기

NewsArticle.kt

```
package com.devtides.androidcoroutinesretrofit.model

import com.google.gson.annotations.SerializedName


data class NewsArticle(
    val author: String? = null,
    val title: String? = null,
    val description: String? = null,
    val url: String? = null,
    @SerializedName("imageUrl")
    val urlToImage: String? = null,
    val publishedAt: String? = null
)
```

NewsService.kt

```
package com.devtides.androidcoroutinesflow.model

import com.devtides.androidcoroutinesretrofit.model.NewsArticle
import retrofit2.http.GET

interface NewsService {
    //https://raw.githubusercontent.com/DevTides/NewsApi/master/news.json

    @GET("news.json")
    suspend fun getNews() : List<NewsArticle>

}
```

NewsRepository.kt

```
package com.devtides.androidcoroutinesflow.model

import com.devtides.androidcoroutinesretrofit.model.NewsArticle
import kotlinx.coroutines.delay
import kotlinx.coroutines.flow.Flow
import kotlinx.coroutines.flow.flow
import retrofit2.Retrofit
import retrofit2.converter.gson.GsonConverterFactory

class NewsRepository{

    companion object{
        private const val BASE_URL =
                 "https://raw.githubusercontent.com/DevTides/NewsApi/master/"
        private const val NEWS_DELAY = 3000L
    }
    private val newsService
    =Retrofit.Builder()
        .baseUrl(BASE_URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build()
        .create(NewsService::class.java)

    fun getNewsArticles() : Flow<NewsArticle>{
        //받아온 객체들을 Flow 로 바꾼다.
        return flow{
            var newsSource = newsService.getNews()
            newsSource.forEach{
                emit(it)
                delay(NEWS_DELAY)
            }
        }
    }
}
```

2. ViewModel 클래스에서 Repository 로 getNews 호출하기

```
package com.devtides.androidcoroutinesflow.viewmodel

import androidx.lifecycle.ViewModel
import androidx.lifecycle.asLiveData
import com.devtides.androidcoroutinesflow.model.NewsRepository

class ListViewModel: ViewModel() {

    val newsArticles = NewsRepository().getNewsArticles().asLiveData()

}
```

- **asLiveData() 매우 편리한 기능인듯!**

3. Repository 에서 받은 정보로 MainActivity 업데이트 하기

```
package com.devtides.androidcoroutinesflow.view

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.View
import androidx.lifecycle.Observer
import androidx.lifecycle.ViewModelProviders
import androidx.recyclerview.widget.LinearLayoutManager
import com.devtides.androidcoroutinesflow.R
import com.devtides.androidcoroutinesflow.viewmodel.ListViewModel
import com.devtides.coroutinesretrofit.view.NewsListAdapter
import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {

    lateinit var viewModel: ListViewModel
    private val newsListAdapter = NewsListAdapter()

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        viewModel = ViewModelProviders.of(this).get(ListViewModel::class.java)

        newsList.apply {
            layoutManager = LinearLayoutManager(this@MainActivity)
            adapter = newsListAdapter
        }

        observeViewModel()
```

```
    }

    private fun observeViewModel() {
        viewModel.newsArticles.observe(this, Observer { article ->
            //Consume the flow
            loading_view.visibility = View.GONE //hiding the spinner
            newsList.visibility = View.VISIBLE
            //Flow 에서 3초마다 뉴스를 받는다.
            newsListAdapter.onAddNewsItem(article)
            newsList.smoothScrollToPosition(0)
        })
    }
}
```

- **새로운 뉴스가 추가될 때마다 스크롤을 가장 첫번째 칸으로 올린다.**