TMDB API

Priority: High Medium Low

GET/search/movie

E.g.
https://api.themoviedb.org/3/search/movie?api_key=dbd468dd443c8122522d47916fbc3a59&query=Justice%20League

Required
api_key : string
query : string (min len 1)

Optional
language : string ("ja-JP", "en-US")
page : int (1-1000)
include_adult : boolean
region : string
year : int

Response
Success 200
Fail 401 404

Model

```
data class MovieSearchResult(
    val page: Int,
    val results: List<Movie>,
    val total_pages: Int,
    val total_results: Int
)

data class Movie (
    val adult: Boolean,
    val backdrop_path: String?,
    val genre_ids: List<Int>,
    val id: Int,
    val original_language: String,
    val original_title: String,
    val overview: String,
    val popularity: Double,
    val poster_path: String?,
    val release_date: String,
```

```
    val title: String,
    val video: Boolean,
    val vote_average: Double,
    val vote_count: Int
)
```

(Note)
Need to found out: max number of result return by one page

GET/movie/{movie_id}

E.g.
https://api.themoviedb.org/3/movie/791373?api_key=dbd468dd443c8122522d47916fbc3a59

Required
api_key : string
movie_id : int

Optional
language : string ("ja-JP", "en-US")
append_to_response : string

Response
Success 200
Fail 401 404

Model

```
data class Genre(
    val id: Int,
    val name: String
)

data class SpokenLanguage(
    val iso_639_1: String,
    val name: String
)

data class ProductionCompany(
    val id: Int,
    val logo_path: String?,
    val name: String,
    val origin_country: String
```

```
)

data class ProductionCountry(
    val iso_3166_1: String,
    val name: String
)

data class MovieDetail(
    val adult: Boolean,
    val backdrop_path: String?,
    val belongs_to_collection: Any?,
    val budget: Int,
    val genres: List<Genre>,
    val homepage: String?,
    val id: Int,
    val imdb_id: String?,
    val original_language: String,
    val original_title: String,
    val overview: String?,
    val popularity: Double,
    val poster_path: String?,
    val production_companies: List<ProductionCompany>,
    val production_countries: List<ProductionCountry>,
    val release_date: String,
    val revenue: Int,
    val runtime: Int,
    val spoken_languages: List<SpokenLanguage>,
    val status: String,
    val tagline: String?,
    val title: String,
    val video: Boolean,
    val vote_average: Double,
    val vote_count: Int
)
```

First Implementation

```
interface TmdbApi {
    @GET("3/search/movie")
    suspend fun searchMovie(@QueryMap params: Map<String, String>):
Response<MovieSearchResult>

    @GET("3/movie/{id}")
    suspend fun movieDetail(@Path(value="id") id:String, @QueryMap params:
Map<String, String>): Response<MovieDetail>
}
```

```
object TmdbApiService {
    val api = create()

    private fun create(): TmdbApi {
        val baseUrl = Common.baseUrl
        val retrofit: Retrofit = Retrofit.Builder().baseUrl(baseUrl)
            .addConverterFactory(MoshiConverterFactory.create())
            .build()
        return retrofit.create(TmdbApi::class.java)
    }
}
```

ApiActions.kt

```
suspend fun actionSearchMovie(query:String, page:Int = 1) :
Response<MovieSearchResult> {
    val params: MutableMap<String, String> = HashMap()
    params["api_key"] = Common.apiKey
    params["language"] = Common.language
    params["page"] = page.toString()
    params["query"] = query

    return TmdbApiService.api.searchMovie(params)
}

suspend fun actionGetMovieDetail(movieId:Int) : Response<MovieDetail> {
    val params: MutableMap<String, String> = HashMap()
    params["api_key"] = Common.apiKey
    params["language"] = Common.language

    return TmdbApiService.api.movieDetail(movieId.toString(), params)
}
```

Test

```
@RunWith(JUnitParamsRunner::class)
class ApiTest {

    fun searchData() = arrayOf(arrayOf("Hero", 200), arrayOf("", 422)) //
response is 422 instead of 404 @ blank
    fun detailData() = arrayOf(arrayOf(791373, 200), arrayOf(-1, 404))

    @Test
    @Parameters(method = "detailData")
    fun testDetailApi(movieId:Int, expectResult:Int) {
        println("Movie Detail API Test")
```

```
        runBlocking {
            val start = Instant.now()

            val job = async { actionGetMovieDetail(movieId) }
            val response = job.await()

            val end = Instant.now()
            println("Response Time: ${Duration.between(start,
end).toMillis()}ms")

            println("Response Code: ${response.code()}")
            if (response.isSuccessful) {
                response.body()?.let {
                    println("Results: $it")
                }
            }

            Assert.assertEquals(expectResult, response.code())
        }
    }

    @Test
    @Parameters(method = "searchData")
    fun testSearchApi(query:String, expectResult:Int) {
        println("Search Movie API Test")

        runBlocking {
            val start = Instant.now()

            val job = async { actionSearchMovie(query) }
            val response = job.await()

            val end = Instant.now()
            println("Response Time: ${Duration.between(start,
end).toMillis()}ms")

            println("Response Code: ${response.code()}")
            if (response.isSuccessful) {
                response.body()?.let {
                    println("Total Pages: ${it.total_pages}")
                    println("Total Results: ${it.total_results}")
                    println("Page Results: ${it.results.size}")
                    println("Results: ${it.results}")
                }
            }

            Assert.assertEquals(expectResult, response.code())
```

```
        }
    }
}
```

Test Result

Search Movie API Test
Response Time: 1274ms
Response Code: 200
Total Pages: 93
Total Results: 1856
Page Results: 20
Results: [Movie(adult=false, backdrop_path=/4s2d3xdyqotiVNHTITlJjrr3q0H.jpg, genre_ids=[12, 10751, 16, 28, 35], id=177572, original_language=en, original_title=Big Hero 6, overview=未来のサンフランソウキョウに住む14歳の少年ヒロ・ハマダは天才少年だが、その才能を非合法のロボット・ファイトに利用するという自堕落な生活を送っていた。そんな弟を見かねた兄のタダシは、彼を自身の所属する工科大学へ連れていく。タダシの友人である「科学オタク」たちの手がけた数々の発明品や、兄の開発した白くて風船の様な見た目のケアロボット「ベイマックス」を目にし刺激を受けたヒロは、科学の夢を追究したいと飛び級入学を決意する。, popularity=97.08, poster_path=/zBf6PwPurXPwnmckH1mETuZa7vQ.jpg, release_date=2014-10-24, title=ベイマックス, video=false, vote_average=7.8, vote_count=12579), Movie(adult=false, backdrop_path=/3vcDoLeEk9EHD9cmcuBLnoGuZtq.jpg, genre_ids=[12, 16, 35, 10751], (---SKIP---)

Search Movie API Test
Response Time: 332ms
Response Code: 422

Movie Detail API Test
Response Time: 46ms
Response Code: 200
Results: MovieDetail(adult=false, backdrop_path=/pcDc2WJAYGJTTvRSEIpRZwM3Ola.jpg, belongs_to_collection=null, budget=70000000, genres=[Genre(id=28, name=アクション), Genre(id=12, name=アドベンチャー), Genre(id=14, name=ファンタジー), Genre(id=878, name=サイエンスフィクション)], homepage=https://www.hbomax.com/zacksnydersjusticeleague, id=791373, (---SKIP---)

Movie Detail API Test
Response Time: 31ms
Response Code: 404

Modify 1

Improve performance - Add error checking to search api

```
suspend fun actionSearchMovie(query:String, page:Int = 1) : Pair<Int,
Optional<MovieSearchResult>> {
    if(page !in 1..1000 || query.isEmpty()) return Pair(404,
Optional.empty())

    val params: MutableMap<String, String> = HashMap()
    params["api_key"] = Common.apiKey
    params["language"] = Common.language
    params["page"] = page.toString()
    params["query"] = query

    val response =  TmdbApiService.api.searchMovie(params)
    return when (response.isSuccessful) {
        true -> Pair(response.code(), Optional.ofNullable(response.body()))
        false -> Pair(response.code(), Optional.empty())
    }
}

suspend fun actionGetMovieDetail(movieId:Int) : Pair<Int,
Optional<MovieDetail>> {
    val params: MutableMap<String, String> = HashMap()
    params["api_key"] = Common.apiKey
    params["language"] = Common.language

    val response = TmdbApiService.api.movieDetail(movieId.toString(),
params)
    return when (response.isSuccessful) {
        true -> Pair(response.code(), Optional.ofNullable(response.body()))
        false -> Pair(response.code(), Optional.empty())
    }
}
```

Test Result

Search Movie API Test
Response Time: 1517ms
Response Code: 200
Total Pages: 93
Total Results: 1856
Page Results: 20
Results: [Movie(adult=false, backdrop_path=/4s2d3xdyqotiVNHTITlJjrr3q0H.jpg, genre_ids=[12, 10751, 16, 28, 35], id=177572, original_language=en, original_title=Big Hero 6, overview=未来
(---SKIP---)

Search Movie API Test
Response Time: 0ms
Response Code: 404