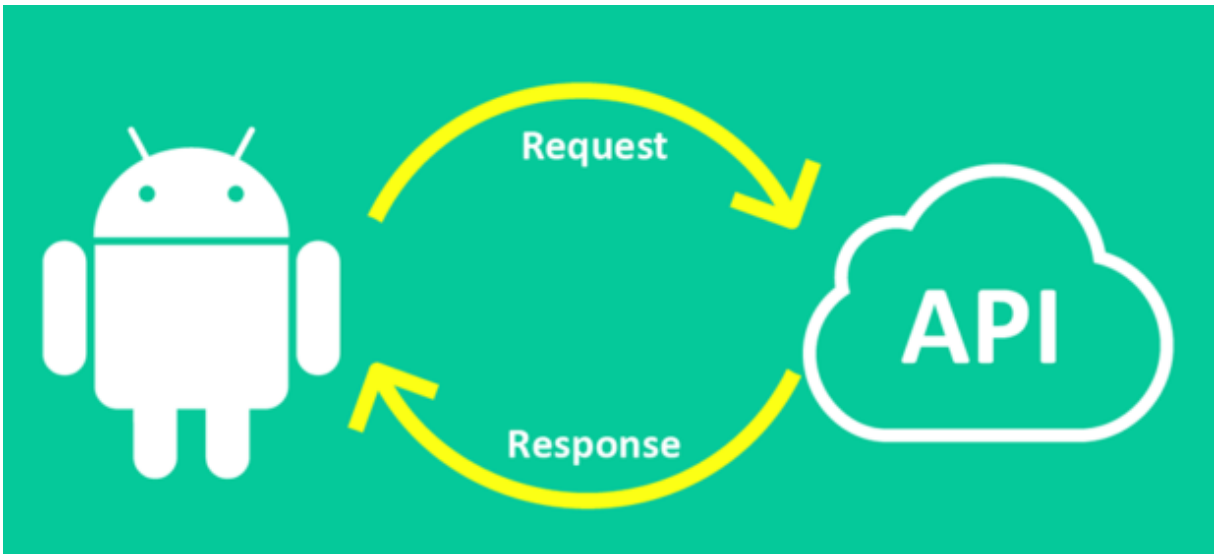
Retrofit Nedir?

Retrofit, Android uygulamalarında RESTful API'lerle iletişim kurmak için kullanılan bir HTTP kütüphanesidir. Gelen veriler default olarak JSON türündedir. Gelen verileri Kotlin/Java class'larına dönüştürmek için Gson kütüphanesi kullanılır.



HTTP Method Çeşitleri:

@GET: Bu metod bir kaynağın okunması, sunucudan veri almak için kullanılır.

```
@GET("/api/v1/employees")
suspend fun getEmployees(): Response<ResponseBody>
```

@POST: Bu metod sunucuya veri göndermek için kullanılır. Örneğin, bir kullanıcı formu gönderirken veya yeni bir blog yazısı oluştururken kullanılabilir.

```
@POST("/api/v1/create")
suspend fun createEmployee(@Body requestBody: RequestBody): Response<ResponseBody>
```

@PUT: Bu metod var olan bilgileri değiştirmek/güncellemek için kullanılır.

```
@PUT("/api/users/2")
suspend fun updateEmployee(@Body requestBody: RequestBody): Response<ResponseBody>
```

@DELETE: Sunucudan/Veri tabanından belirli bilgiyi silmek için kullanılır.

```
@DELETE("/typicode/demo/posts/1")
suspend fun deleteEmployee(): Response<ResponseBody>
```

Retrofit içerisinde en çok kullanılan Annotation'lar

@Path: URL üzerinde bir değişken belirtmek için kullanılır. Eğer @GET kullanırken spesifik bir bilgi üzerine işlem yapılacaksa @Path kullanılabilir.

```
@GET("/api/users/{Id}")
suspend fun getEmployee(@Path("Id") employeeId: String): Response<ResponseBody>
```

Bu kodda @GET içinde bir "Id" belirtilmiş. @Path kullanarak metodun parametresi olan "employeeId" @GET içindeki "Id"nin yerini tutar.

@Body: Parametre olarak bir nesne ilettiğimizde @Body kullanılır.

```
@PUT("/api/users/2")
suspend fun updateEmployee(@Body requestBody: RequestBody): Response<ResponseBody>
```

Kodda @Body olarak bir RequestBody nesnesi iletilmiştir.

@Query: @GET isteği yaparken URL'deki değişkeni belirtmek için kullanılır. Örneğin şöyle bir URL olsun: **/api/users?page=2** buradaki "page" değişkenini @Query aracılığıyla iletiriz. Aşağıdaki kodda örnek bir kullanım verilmiştir.

```
@GET("/api/users")
suspend fun getEmployees(@Query("page") page: String?): Response<ResponseBody>
```

HTTP Status Kodları:

HTTP status kodları isteklerin sonuçlarını ve alınan hataların anlanmasına olanak sağlar.

Bu status kodlarını şöyle ayırabiliriz:

- 1_ _(**Informational**)
  - 100 (Continue): Gönderilen isteğin sunucu tarafından alındığını ve veri gövdesinin de alınmaya hazır olduğunu bildirir.
  - 101 (Switching Protocols): Sunucunun yeni bir protokole geçiş yaptığını belirtir.
- 2_ _(**Success**)
  - 200 (OK): İsteğin başarılı şekilde tamamdandığını bildirir.
  - 201 (Created): Sunucu Tarafından alınan isteğin tamamlanıp bir kaynak oluşturulduğunu belirtir.
  - 204 (No Content): İsteğin başarılı olup geriye bir yanıt dönmediği durumu belirtir.
- 3_ _(**Redirection**)
  - 301 (Moved Permanently): Kaynağın kalıcı olarak başka bir adrese taşındığını belirtir.
  - 302 (Found): Kaynağın geçici olarak başka bir adrese taşındığını belirtir.
- 4_ _(**Client Error**)
  - 400 (Bad Request): İstek formatının hatalı ya da sunucu tarafından anlaşılamadığını belirtir.
  - 401 (Unauthorized): Erişilmek istenilen kaynağın geçerli kimlik bilgilerince erişilmesi gerektiğini belirtir. Geçersiz bilgiler denenirse bu kod karşımıza çıkar.
  - 403 (Forbidden): Kaynağa erişimin yasak olduğunu belirtir.
  - 404 (Not Found): Kaynağın sunucuda olmadığını belirten koddur.
  - 429 (Too Many Requests): Sunucya çok fazla istek gönderildiğinde sunucu tarafından istek sınırı getirildiğini belirtir.
- 5_ _(**Server Error**)
  - 500 (Internal Server Error): Sunucu içinde hata olduğunu belirtir.
  - 502 (Bad Gateway): Sunucunun başka bir sunucuya istek gönderdikten sonra geçersiz/hatalı yanıt alması durumunu belirtir.

- 503 (Service Unavailable): Sunucunun geçici olarak hizmet vermediğini ifade eder. (Bakım/Aşırı yüklenme durumlarında)

Yusuf Mücahit Solmaz