커머스 앱 - 메인 매장 화면 개발하기

6 ViewModule 리스트 개발-injectable & page refresh

injectable (ver 2.2.0 강의 기준)

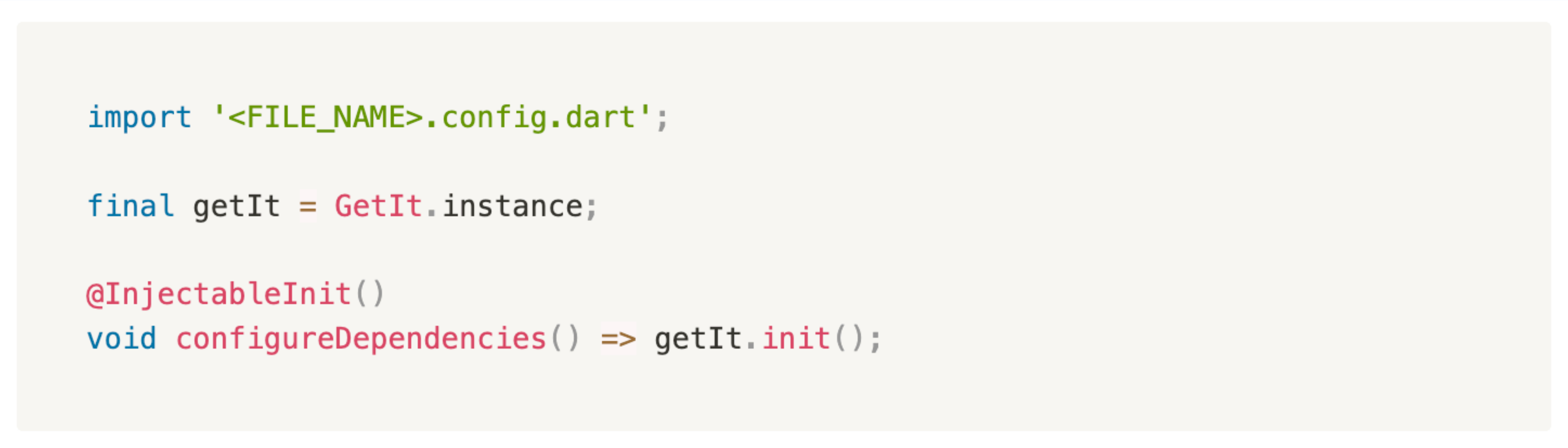
https://pub.dev/packages/injectable

Get It의 의존성 등록 기능을 **Annotation 기반 Code Generation**를 통해 **자동화** 시켜주는 패키지

```
$  flutter pub add injectable
$  flutter pub add -d injectable_generator
```

Injectable - set up

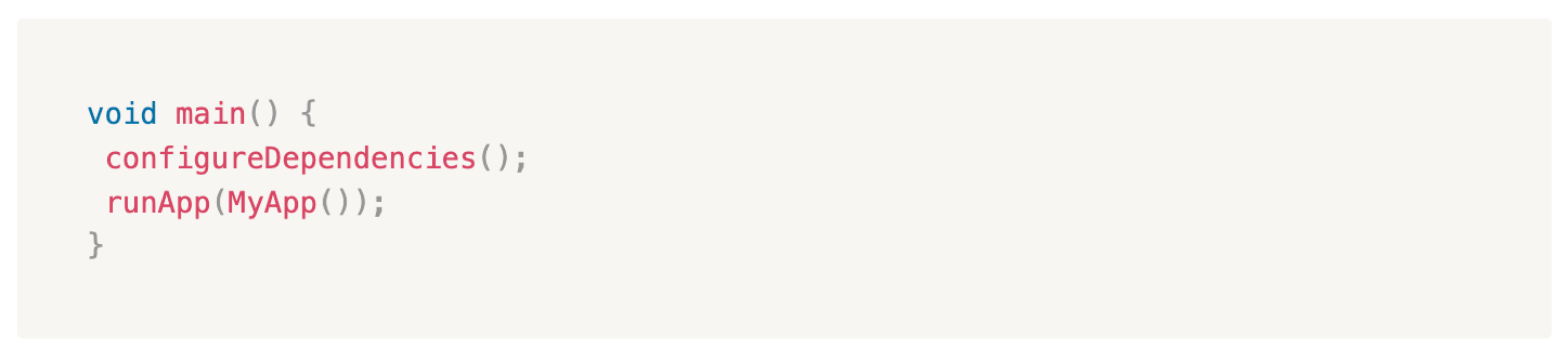
Set up



```dart
import '<FILE_NAME>.config.dart';

final getIt = GetIt.instance;

@InjectableInit()
void configureDependencies() => getIt.init();
```

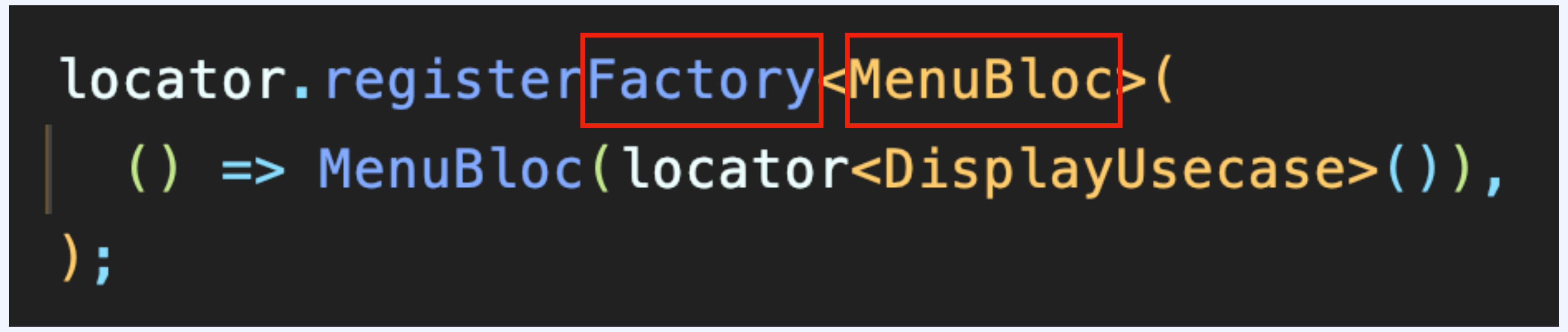


```dart
void main() {
 configureDependencies();
 runApp(MyApp());
}
```

Injectable - 기능 알아보기

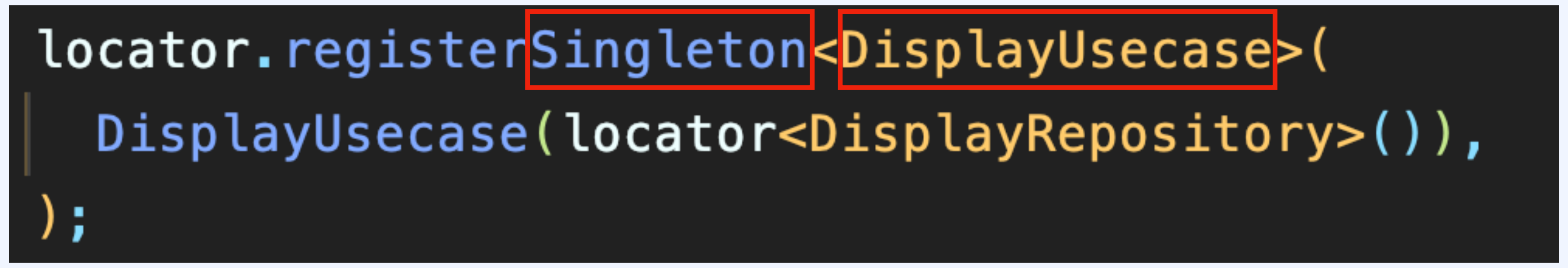
1. 의존성 등록

(1) factory

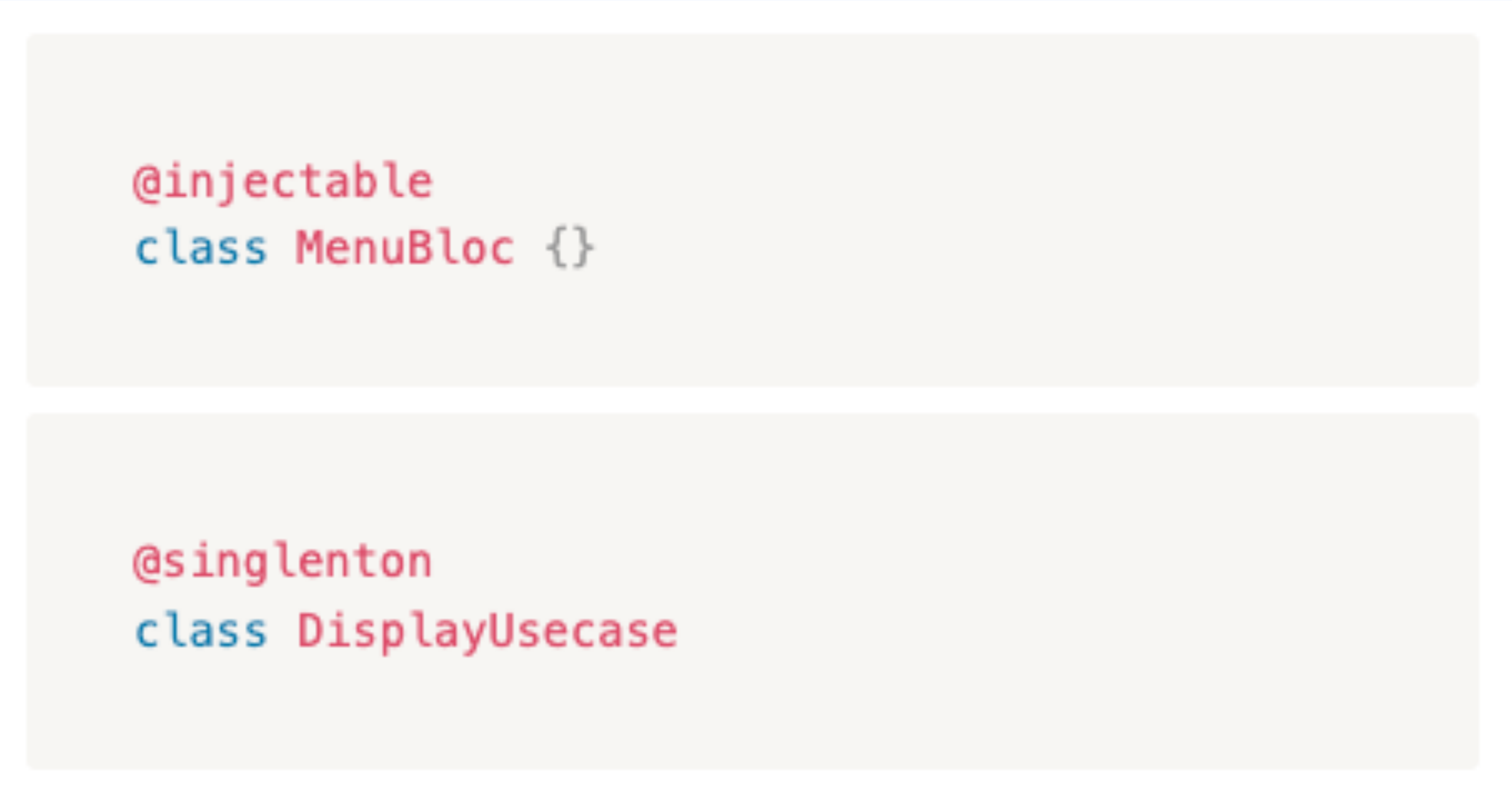


```
locator.registerFactory<MenuBloc>(
  () => MenuBloc(locator<DisplayUsecase>()),
);
```

(2) singleton



```
locator.registerSingleton<DisplayUsecase>(
  DisplayUsecase(locator<DisplayRepository>()),
);
```

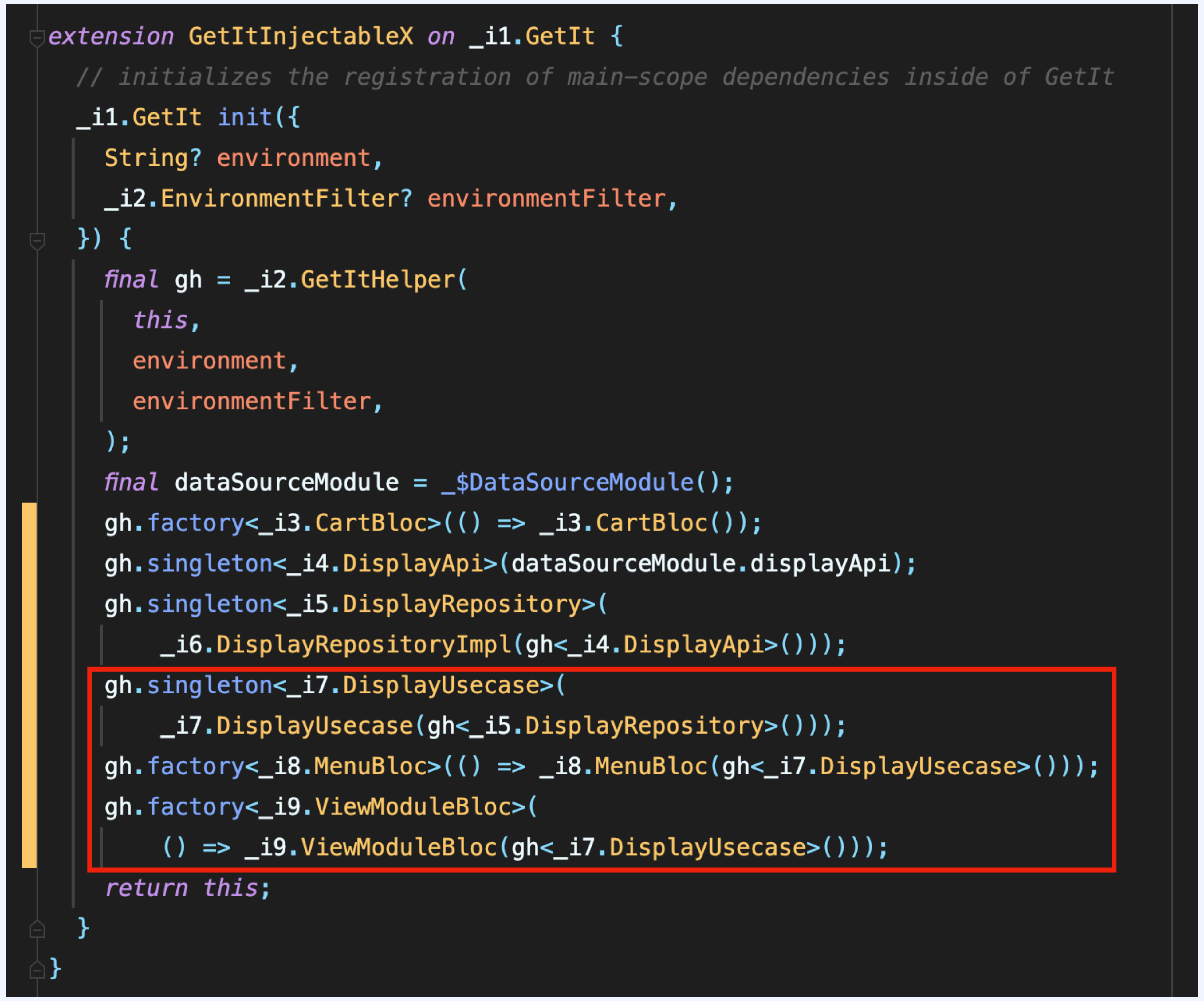


```
@injectable
class MenuBloc {}
```

```
@singleton
class DisplayUsecase
```

Injectable - 기능 알아보기

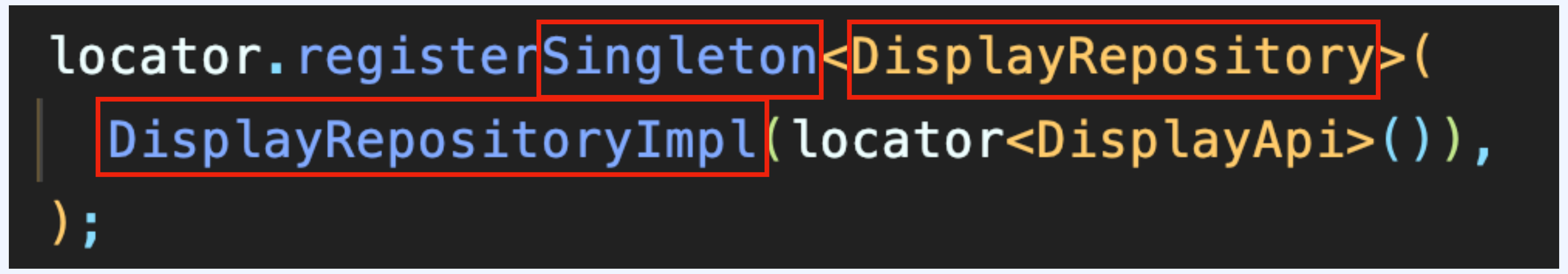
1. 의존성 등록



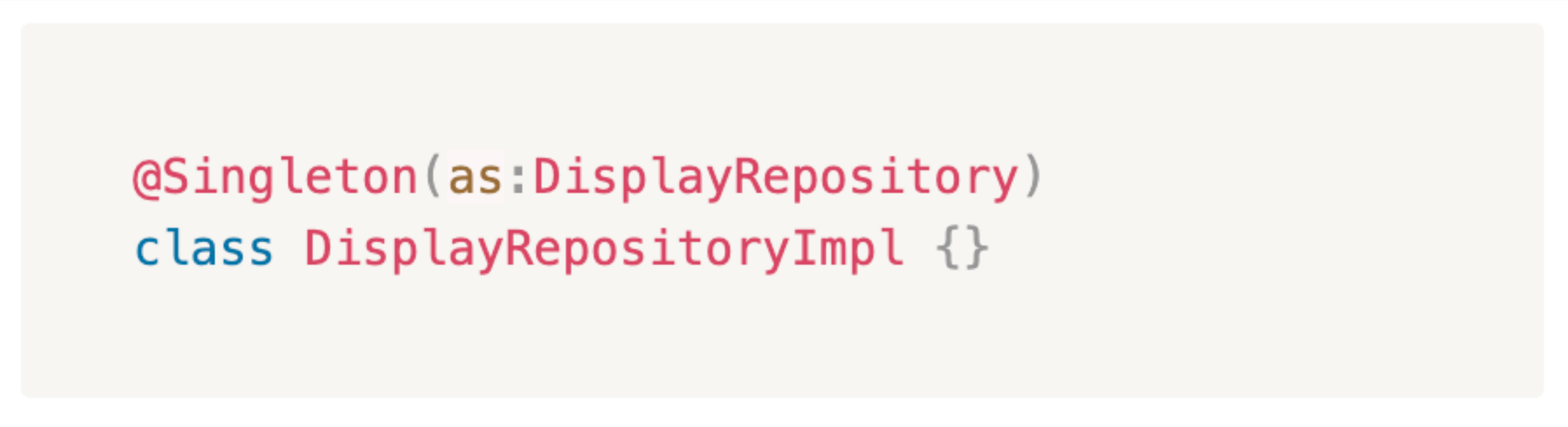
```
extension GetItInjectableX on _i1.GetIt {
  // initializes the registration of main-scope dependencies inside of GetIt
  _i1.GetIt init({
    String? environment,
    _i2.EnvironmentFilter? environmentFilter,
  }) {
    final gh = _i2.GetItHelper(
      this,
      environment,
      environmentFilter,
    );
    final dataSourceModule = _$DataSourceModule();
    gh.factory<_i3.CartBloc>(() => _i3.CartBloc());
    gh.singleton<_i4.DisplayApi>(dataSourceModule.displayApi);
    gh.singleton<_i5.DisplayRepository>(
        _i6.DisplayRepositoryImpl(gh<_i4.DisplayApi>()));
    gh.singleton<_i7.DisplayUsecase>(
        _i7.DisplayUsecase(gh<_i5.DisplayRepository>()));
    gh.factory<_i8.MenuBloc>(() => _i8.MenuBloc(gh<_i7.DisplayUsecase>()));
    gh.factory<_i9.ViewModuleBloc>(
        () => _i9.ViewModuleBloc(gh<_i7.DisplayUsecase>()));
    return this;
  }
}
```

Injectable - 기능 알아보기

2. 추상화 클래스와 구현 클래스의 연결



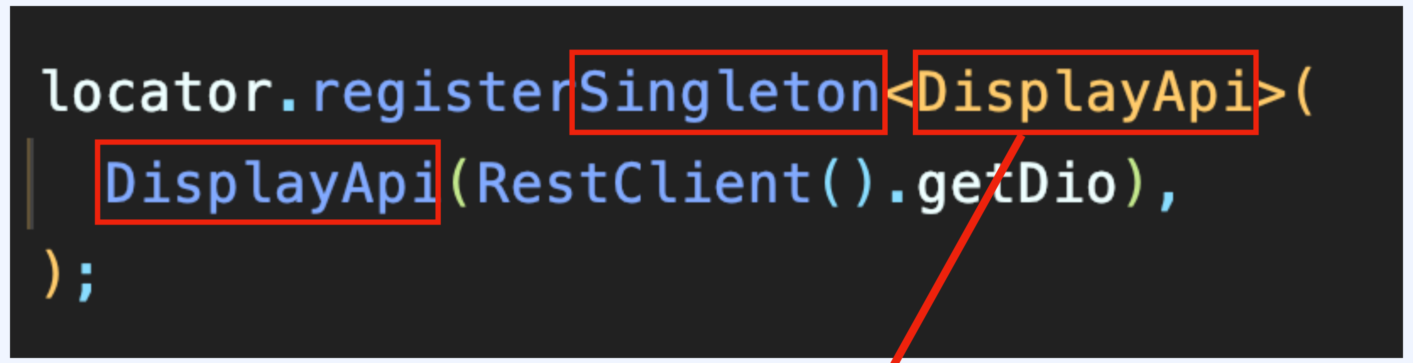
```
locator.registerSingleton<DisplayRepository>(
  DisplayRepositoryImpl(locator<DisplayApi>()),
);
```



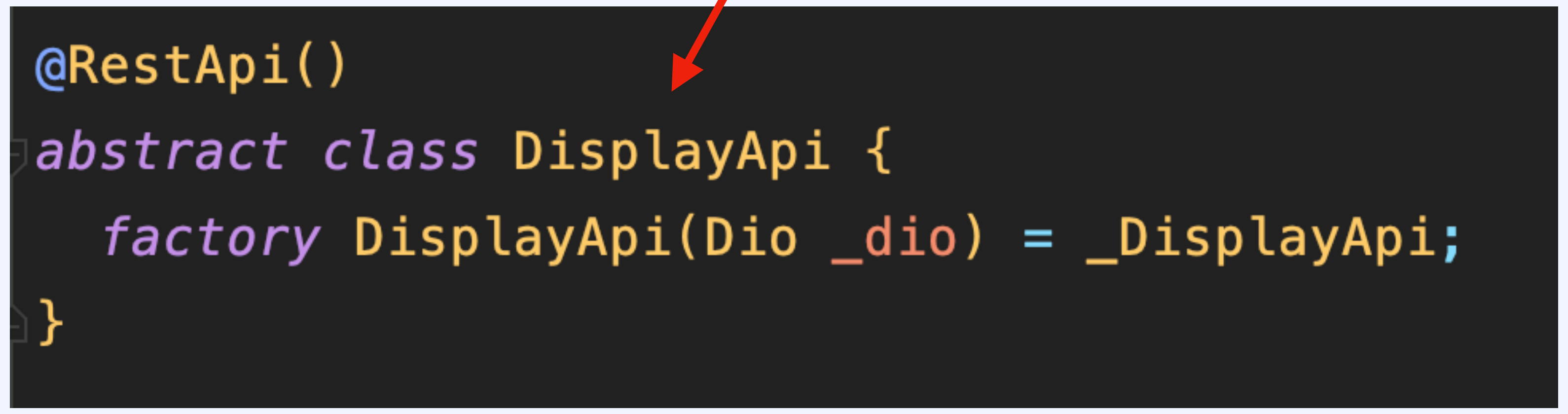
```
@Singleton(as:DisplayRepository)
class DisplayRepositoryImpl {}
```

Injectable - 기능 알아보기

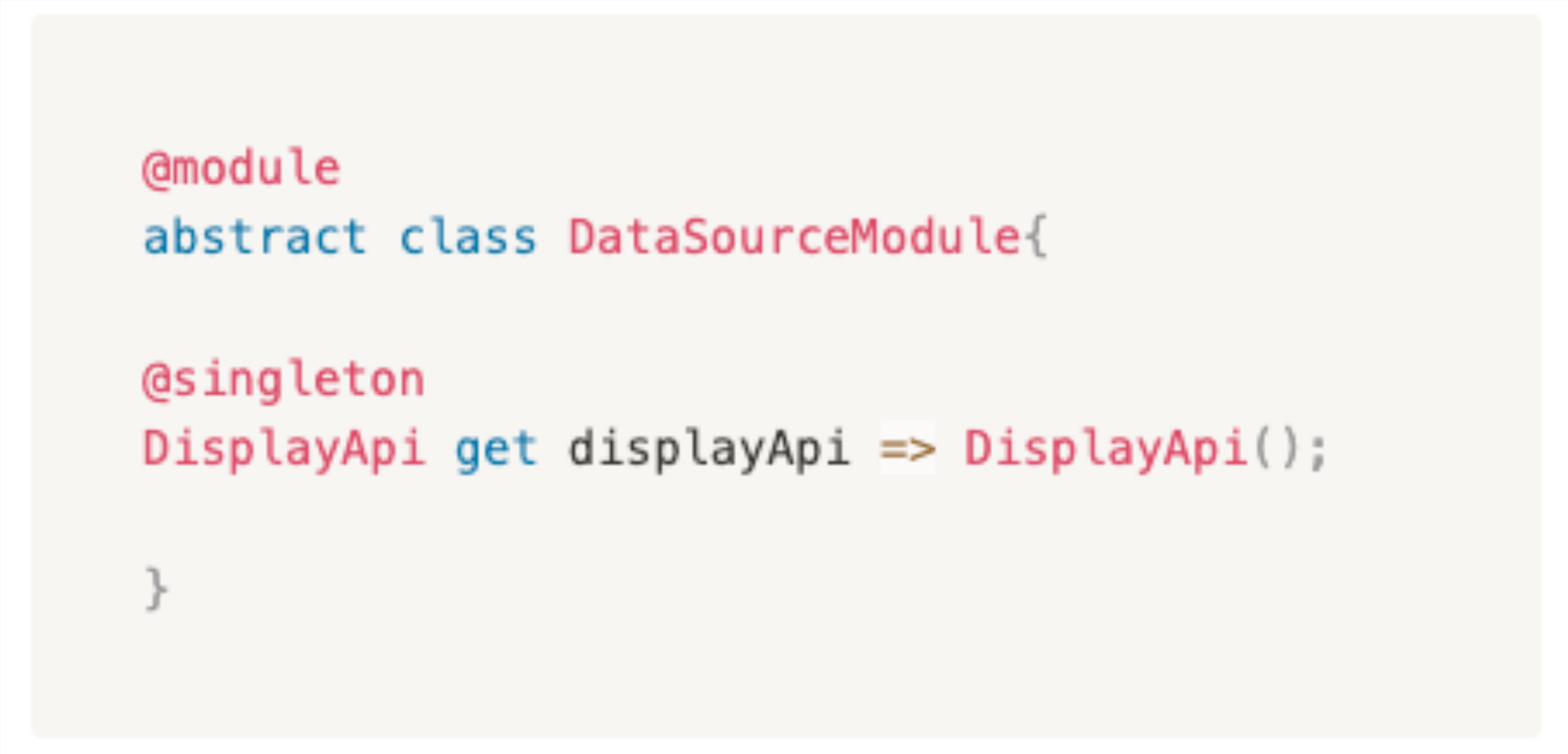
3. Third party types 등록



```
locator.registerSingleton<DisplayApi>(
  DisplayApi(RestClient().getDio),
);
```



```
@RestApi()
abstract class DisplayApi {
  factory DisplayApi(Dio _dio) = _DisplayApi;
}
```



```
@module
abstract class DataSourceModule{

@singleton
DisplayApi get displayApi => DisplayApi();

}
```

Let’s start coding!