KONU: Dart ile programlama temelleri.

Repository: "https://github.com/BerkeCanGoktas/Staj"

Not: Flutter ile programlama için flutterın, visual studio code ve visual studio code flutter extensionının ayrıca da android studio sdksinin bilgisayarınızda kurulu olması gerekmektedir. "https://docs.flutter.dev/get-started/install/windows" linkindeki adımlar takip edilerek kurulum gerçekleştirilebilir.

Not: Dart ile programlamaya girişten önce temel programlama mantığına hâkim olmak önemlidir. Bu raporda dartın C++tan ayrılan yönleri ele alınmıştır.

Exceptions

Dart exception handlingi temelde tüm dillerde olduğu gibidir. Farklı olarak on Exception catch ile birden fazla catch durumu ele alınabilir ve rethrow keywordü ile bir exception parçalı olarak ele alınabilir.

Bir Objenin Türünü Öğrenme

Bir objenin türü runtimeType metodu ile öğrenilebilir ancak is keywordü ile kontrol sağlamak daha doğru bir kullanımdır.

Redirecting Constructors

Bir named constructor, : this(params) şeklinde bir tanımlama ile sınıfın default constructorını çağırabilir. Bu tür constructorlara redirecting constructor denir.

Constant Constructors

Bir sınıfın objeleri programın ömrü boyunca değişikliğe uğramıyorsa, bu sınıfın constructorı const olarak tanımlanabilir. Bu tanımlamanın hatasız olması için sınıfın tüm üyeleri final keywordü ile kullanılmalıdır.

Factory Constructors

Bir constructor factory keywordünü alarak sınıfın objesinden başka bir tür de döndürebilir. Örneğin bu yöntemle bir sınıf subclassının bir objesini döndürebilir.

noSuchMethod()

Kod tanımlanmamış/var olmayan bir fonksiyonu çağırmaya çalıştığında bu metot çağrılır. noSuchMethod() override edilerek böyle bir durumla karşılaşıldığında programın yapmasını istediği davranış belirtilebilir.

Mixins

Dart multiple inheritence destekleyen bir dil değildir. Ancak mixin konsepti sayesinde sınıfların özellik ve üyeleri genişletilebilir. mixin Mixin_Name ile eklenti tanımlanır. mixin Mixin_Name on Class_Name ile sadece belli bir sınıfın subclassları için mixin tanımlaması yapılabilir. Bir mixini classta kullanmak için ise class tanımlamasına (süslü parantezlerden önce) with Mixin_Name eklenmelidir.

Asynchrony

Bir fonksiyonun tanımlamasına (süslü parantezlerden önce) async keywordü eklenirse o fonksiyon asenkron şekilde çalışabilir. Asenkron çalışacak satırlar await keywordü ile belirtilir. Asenkron fonksiyonlar aynı zamanda senkron satırlar da içerebilir (normalde yazdığımız kodlar senkrondur). Önemli bir nokta da bu fonksiyonların return typelarıdır. Asenkron fonksiyonlar Future objesi döndürür. Future objesi programa bir değer döndürüleceğinin vaadini verir. İşlem tamamlanmışsa değer veya error, devam ediyorsa uncompleted Future objesi döndürülür. Errorler exception handling ile denetlenebilir.

Not: Main fonksiyonu da asenkron olabilir.

Callable Classes

Bir sınıfta call adında bir fonksiyon tanımlanabilir. Eğer bu tanımlama yapılırsa bu sınıftan türetilen objeler call fonksiyonuna verilen parametreler ile çağrılabilir ve bu fonksiyonun yapacağı işlemi gerçekleştirir.

Metadata

@ ile metadata bilgiler yazılabilir, programın çalışmasına etki etmez ancak okunabilirliği açısından önemlidir. Örneğin subclassta override edilen bir fonksiyonun başına @override metadatası konulabilir.

Paket Ekleme

Paketler projeye esneklik katar. Örneğin normalde dartta xmlden jsona dönüşüm için fonksiyonlar mevcut değildir. Ancak xml2json paketi ile bu işlevsellik yazdığımız programa kazandırılabilir. Dart/flutter projelerine paket eklemek oldukça basittir. Bunun için projemizin dosyasında komut istemine girip dart pub add <paket_adı> veya (flutter ile çalışıyorsak) flutter add <paket_adı> komutunu çalıştırmak yeterlidir. Arzuladığımız paketlere ise istediğimiz paket adını "https://pub.dev/packages/xml2json/install" adresinde aratarak ulaşabilir ve inceleyebiliriz.

Berke Can GÖKTAŞ

01.12.2021