FLUTTER YORDAMIDA PLATFORMALARARO KUTUBXONANI ISHLAB CHIQISH TEXNOLOGIYASI

Farmonov Sherzodbek Raxmonjonovich
Amaliy matematika va informatika kafedrasi katta o‘qituvchisi
Farg‘ona davlat universiteti
Rahmatjonov Mirzohidjon Muzaffarjon o’g’li
Amaliy matematika va informatika yo‘nalishi 4-kurs talabasi
Farg‘ona davlat universiteti
https://doi.org/10.5281/zenodo.8020544

ARTICLE INFO
Qabul qilindi: 01-June 2023 yil
Ma’qullandi: 05-June 2023 yil
Nashr qilindi: 09-June 2023 yil
KEY WORDS
Dart, Flutter, Dio, Retrofit, React Native, Android, iOS, JSON.

ABSTRACT
Mazkur maqolada flutter texnologiyasi yordamida platformalararo kutubxonalar ishlab chiqishni ko’rib chiqadi. Flutter - bu bir xil kod bazasiga ega iOS va Android qurilmalari uchun yuqori unumdorlik, yuqori aniqlikdagi mobil ilovalarni ishlab chiqish uchun ochiq manbali SDK. Biz bu maqola yordamida flutter haqida chuqurroq bilimga ega bo‘lami.

Dart platformasi orqali Android va iOS tizimlari uchun mobil ilovalar ishlab chiqishning eng mashhur usuli "Flutter" hisoblanadi. Flutter, Google tomonidan ishlab chiqilgan bir kross-platforma frameworkidir. U foydalanuvchiga Dart dasturlash tilini ishlatish orqali Android va iOS uchun bir nechta platformalararo mobil ilovalar yaratish imkonini beradi.

Flutter orqali mobil ilova yaratish jarayoni quyidagi bosqichlarga mos keladi:

1. Flutter va Dart o'rnatish: Flutter SDK ni o'rnatilishi kerak, shuningdek Dart ni ham o'rnatish kerak. Flutter SDK, ilovalarni yaratish vaqti hisobga olinganda qo'llaniladigan usullarni va resurslarni o'z ichiga oladi.
2. Yangi proyekt yaratish: Flutter proyektlarini yaratish uchun "flutter create" komandasini ishga tushiriladi:

flutter create *proyekt_nomi*

3. Ilova prototipi va interfeysi: Flutter ilovalari uchun prototip yoki interfeysni quradi. Flutter uchun "Widget" lar hierarxini ishlatib, ilova elementlarini va shakllarni yaratadi.
4. Biznes logikasini va funktsionalni yozish: Dart tilida ilova funktsionali va biznes logikasini yoziladi. Dart tilida funksiyalar, klasslar, interfeyslar va boshqa modullar yaratish mumkin.
5. Ilova interfeysini shakllantirish: Flutterning shakllantirish oynasida (android emuliator) da ilovaning interfeysini tasvirlash mumkin. Flutterning qo'llaniladigan vidjetlari va qo'llanmalaridan foydalanib, ilovaning qulay va ko'rsatiladigan interfeysini yaratish mumkin.
6. Test qilish: Ilovaning to'g'ri ishlashini tekshirish uchun testlar yoziladi. Flutter, testlarni avtomatlashtirish uchun vidjetlarni tekshirish, integratsiya va birlikdagi testlar

uchun xususiy tahlillash jarayonlarini taqdim etadi.

7. Android va iOS uchun ilovani build qilish(qurish): Flutter orqali Android va iOS uchun ilovangizni build qilib olinadi. Bu foydalanuvchiga APK va IPA fayllarini yaratish imkonini beradi.

8. Play Market va App Store uchun ilovani joylash: Ilovangizni Play Market (Android) va App Store (iOS) da joylashtirish jarayonlarini amalga oshirish. Bu jarayonlarning qoidalari va talablari bilan tanishib chiqish kerak.

Flutter platformasi, Android va iOS tizimlarida qo'llaniladigan mobil ilovalarni yaratishni juda osonlashtiradi. Uning yaxshi samaradorlik, ma'noli animatsiya va dizayn imkoniyatlari mavjud. Shuningdek, Flutter jamoatchilik tomonidan keng qo'llaniladigan va dolzarb tahlil olishi oson bo'lgan bir frameworkdir.

Dart dasturlash tili platformasida bog'lamalar (bindings) bilan ishlash uchun "ffi" nomli kutubxona (library) mavjud. FFI (Foreign Function Interface) kutubxonasini ishlatib, Dart tilidagi kodni boshqa dasturlash tillariga (masalan, C, C++ yoki Objective-C) bog'lash imkoniyatiga ega bo'lish mumkin.

Dart tilida bog'lamalar bilan ishlash uchun quyidagi jarayonlarni amalga oshirishingiz kerak:

- Bog'lamalar uchun interfeysni aniqlang: Bog'lamalar orqali aloqa o'rnatmoqchi bo'lgan dasturlash tillarining interfeysini (API) aniqlang. Bu interfeys, bog'lamalarda ishlatiladigan funksiyalar, strukturalar va konstantalar to'plamini o'z ichiga oladi.
- Bog'lamalarni yaratish: Bog'lamalar uchun qo'llanmalar va funksiyalar bilan ishlash uchun bog'lanishni yaratish lozim. Bu jarayonda "ffi" kutubxonasi orqali Dart tilida bog'lanishlar, funksiyalar, argumentlar va qaytarish qiymatlari aniqlanadi.
- Bog'lamalarni chaqirish: Aniqlangan bog'lamalarni ishlatib, bog'langan dasturlash tillaridagi funksiyalarni Dart tilida chaqirish mumkin. Bog'lanishlar orqali dastur ichidagi boshqa tillardagi funksiyalarni chaqirish imkoniyatiga ega bo'linadi.
- Xotira boshqaruvini amalga oshirish: Bog'lamalar bilan ishlashda xotira boshqaruvini to'g'ri boshqarish juda muhimdir, xususan, Dart va boshqa tillar orasida ma'lumot almashish paytida. Xotira to'g'ri ravishda ajratilishi va ozod qilinishi uchun to'g'ri usullarni ishlatish kerak.
- Xatoliklar bilan ishlash: Bog'lamalar bilan ishlash jarayonida xatolar bilan to'g'ri kelishish muhimdir. Bog'lanishlarda va bog'lanishlar bilan ishlayotgan funksiyalarda yuz beradigan xatolarni aniqlab, ulardan to'g'ri kelishish uchun moslashtirishlar o'tkaziladi.

Bog'lamalar – Dart platformasida boshqa dasturlash tillariga bog'lashni ta'minlaydi va unga qo'shimcha imkoniyatlar yaratadi. Ammo bog'lamalar bilan ishlash jarayoni odatiy dasturlashdan bir ozodlik va ehtiyotkorlik talab etadi. Bog'lamalar bilan ishlashda amalga oshirish paytida ehtiyot bo'lish kerak va xotira boshqaruvini to'g'ri bajarishga intilish muhim.

Flutter ilovalarida RESTful API-lar bilan ishlash. Flutter ilovalarida RESTful API-lar bilan ishlashni osonlashtirish uchun "http" kutubxonasini ishlatishi mumkin. Bu kutubxona, HTTP-klient funktsiyalarini o'z ichiga oladi va RESTful API-lar bilan kommunikatsiyani osonlashtiradi.

Quyidagi jarayonlarni amalga oshirib Flutter ilovangizda RESTful API-lar bilan ishlash mumkin:

- Kutubxonani o'rnatish: Ilk navbatda, proyektga "http" kutubxonani qo'shiladi.

pubspec.yaml faylida dependencies bo'limiga quyidagi qatorni qo'shib qo'yiladi:
dependencies: http: ^0.13.3
Keyin terminalda flutter pub get komandasini ishga tushirilsa, kutubxona o'rnatiladi.

- API-ga so'rov yuborish: RESTful API-ga so'rov yuborish uchun http kutubxonasidagi get, post, put, delete funktsiyalaridan foydalaning. Misol uchun, GET so'rovi yuborish uchun:

```
import 'package:http/http.dart' as http;
void fetchUsers() async {
  var url = Uri.parse('https://api.example.com/users');
  var response = await http.get(url);
  if (response.statusCode == 200) {
    // Ma'lumotlar qabul qilindi
    var data = response.body;
    // JSON ma'lumotlarni tahlil qilish o'rniga kerakli amallarni bajaring
  } else {
    // Xatolik sodir bo'ldi
    print('Xatolik: ${response.statusCode}');
  }
}
```

- Javoblarni qabul qilish: API-dan kelgan javoblarni tahlil qilib kerakli amallarni bajariladi. Agar API JSON formatida javob qaytarsa, dart:convert kutubxonasi bilan JSON ma'lumotlarini tahlil qilish mumkin:

```
import 'dart:convert';
// ...
void fetchUsers() async {
  // ...
  if (response.statusCode == 200) {
    var data = json.decode(response.body);
    // JSON ma'lumotlarni tahlil qilish va ulardan foydalanish
  } else {
    // ...
  }
}
```

- So'rovga ma'lumotlar yuborish: Agar POST, PUT yoki boshqa HTTP so'rovlari bilan ma'lumot yuborish kerak bo'lsa, http kutubxonasi orqali ma'lumotlarni yuborish imkoniyatiga ega bo'lasiz:

```
void createUser() async {
  var url = Uri.parse('https://api.example.com/users');
  var body = json.encode({'name': 'John', 'email': 'john@example.com'});
  var headers = {'Content-Type': 'application/json'};
  var response = await http.post(url, body: body, headers: headers);
  // ...
}
```

Bu usul orqali Flutter ilovalarida RESTful API-lar bilan ishlash osonlashtiriladi. To'g'ri URL-ni va kerakli metodlarni tanlang va HTTP-klient funktsiyalaridan foydalaning. Agar API-

da avtorizatsiya kerak bo'lsa, headerlarni sozlash va ma'lumotlarni yuborishda foydalanishingiz mumkin.

Hulosa qilib aytganda, Flutter yuqori mahsuldorlik va silliq animatsiyani ta'minlaydi, shuningdek, mobil ilovalarni yaratish uchun ko'plab tayyor vidjetlar va kutubxonalarga kirishni ta'minlaydi. Bundan tashqari, u faol dasturchilar hamjamiyatiga va boy kutubxonaga ega bo'lib, bu esa ishlab chiqish jarayonini oson va samaraliroq bo'lishi xizmat qiladi.

Foydalanilgan adabiyotlar ro'yhati:

1. Tyagi P. Pragmatic Flutter: Building Cross-Platform Mobile Apps for Android, iOS, Web, & Desktop. 1st ed. Boca Raton: CRC Press; 13 August 2021.
2. Hacernoon.com. what's Revolutionary about Flutter [Internet]. 2017. Available https://hackernoon.com/whats-revolutionary-about-flutter946915b09514.
3. Hoang Ly. State Management Analyses of the Flutter Application. BSc. Thesis. Metropolia University of Applied Sciences; 2019.
4. Fayzullaev J. Native-like Cross-Platform Mobile Development MultiOS Engine & Kotlin Native vs Flutter. BSc. Thesis. South Eastern Finland University of Applied Sciences; 2018.
5. Fentaw AE. Cross platform mobile application development: a comparison study of React Native Vs Flutter. MSc. Thesis. University of Jyvaskyla; 2020.
6. Kuitunen M. CROSS-PLATFORM MOBILE APPLICATION DEVELOPMENT WITH REACT NATIVE. BSc. Thesis. Tampere University of Technology; 2018.
7. Docs.flutter.dev. Developing packages & plugins. [Internet]. Available https://docs.flutter.dev/development/packages-andplugins/developing-packages.
8. CompanionLink Blog. The Benefits of Using APIs in Mobile App Development [Internet]. 2021. Available from: https://www.companionlink.com/blog/2021/02/the-benefits-of-usingapis-in-mobile-app-development/.
9. Mamoun R, Nasor M, Abulikailik SH. Design and Development of Mobile Healthcare Application Prototype Using Flutter. In2020 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE) 2021 Feb (pp. 1-6). IEEE. Doi:10.1109/ICCCEEE49695.2021.9429595.
10. Фармонов, Ш., & Камбарова, Д. (2022). КАК ПОМОЧЬ УЧЕНИКАМ РАЗВИТЬ ИНТЕРЕС К УЧЕБЕ. O'rta Osiyo ta'lim va innovatsiyalar jurnali, 1(2), 118-120.
11. Tojiyev, T., Boynazarov, A., & Farmonov, S. (2022). PHARMACOKINETICS IS A DESCRIPTION OF DRUGS AND THEIR BEHAVIOR IN THE HUMAN BODY BY BUILDING A MATHEMATICAL MODEL. Eurasian Journal of Medical and Natural Sciences, 2(13), 146-149.