MODEL VIEW PRESENTER (MVP) ARXITEKTURASI

Mahkamov Shohruh Sarvar o'g'li

Mirzo Ulug'bek nomidagi O'zbekiston milliy universiteti Jizzax filiali

mahkamov@jbnuu.uz

Annotatsiya: *Model View Presenter (MVP) arxitekturasi Android dasturlashda juda mashhur. Bu arxitektura, dasturlash loyihasini qisqa va aniq bo'lishi, kodni test qilishni osonlashtirish, va interfeysni modeldan ajratishni maqsad qiladi. Bu arxitekturada, komponentlar haqida to`liq ma'lumotlar keltirib o'tilgan.*

Kalit so'zlar: *Presenter, Arxitektura (Architecture), Model, View, Foydalanuvchi Interfeysi (User Interface), So'rov (Query), Murojaat (Request), Ma'lumot (Data), Bog'lanish (Binding), Test (Testing).*

Androidni ishlab chiqishning boshlang'ich bosqichida, o'rganuvchilar kodlarni shunday yozadilarki, dasturning barcha logik jarayonlari MainActivity klassida joy oladi. Ilovalarni ishlab chiqishning bunday yondashuvi Android activity ning UI ni ham, dastur ma'lumotlarini qayta ishlash mexanizmini ham bir biriga yaqinlashishiga olib keladi. Bunday mobil ilovalarni saqlash va kengaytirishda ham qiyinchiliklar tug'iladi. Dasturlarning saqlanishi, o'qilishi, kengaytirilishi va qayta ishlab chiqishida uchraydigan bunday muammolarni oldini olish uchun dasturchilar kodni qatlamlarga ajratishni afzal ko'rishadi. Dasturiy arxitektura patternlarni qo'llagan holda, dastur kodini tartibga solish uchun vazifalarni bo'limlarga ajratish lozim[1].

Ushbu maqolada keltiriladigan Model-view-presenter (MVP) haqidagi ma'lumotlarga to'xtalishdan oldin arxitektura tushunchasiga ozroq ta'rif berib o'tsak. Agar biz dasturni bir nechta qoidalarga asoslanib qurayotgan bo'lsak, tegishli funksiyalarni ishlatib va ma'lum bir protokollar bilan ularni ishga tushirsak buni biz "Arxitektura" deb nomlashimiz mumkin.

Aytaylik, agar biz biror bir arxitekturadan foydalanmasak va o'z kodimizni class, activity, yoki fragmentda tartibsiz ravishda yozsak, u holda biz quyidagi

muammolar duch kelishimiz mumkin:

- Tushunish qiyin bo’lgan kodlarning qatorlari ko’payib borishi
- kodni o’qishni yanayam qiyinlashtiribgina qolmay, xatolarni sonini ham bir munchaga oshirib yuborish mumkinligi bunda dasturni test qilish oson bo’lmasligi va bu mahsulotni sifatini tushirishga olib keladi.

Demak aniq bir ma’lumotlar oqimini ta’minlash, mustahkamlilik, o’lchovlilik, xatolarga chidamlilik, tushunarlilik, o’zgartirishlilikni oson va samaradorlikni oshirish bilan birga, sifatli dasturni ham taqdim etadi[2].

Dasturiy ta’minotning Arxitektura Patternlarini dasturlashni targ’ib qiladi. Test qilish oson bo’lgan va arzon narxlardagi texnik xizmat ko’rsatishni ta’minlaydigan dastur funksiyalarini ajratib turadi. Hozirgi kunda eng mashhur arxitektura patternlarga MVC, MVP, MVVM larni misol qilib olishimiz mumkin. Keling, ulardan biri MVP haqida to`liqroq ma’lumot berib o`tamiz.

Model-view-presenter dasturiy ta'minoti namunasi 1990-yillarning boshida Apple, IBM va Hewlett-Packard qo'shma korxonasi Taligentda paydo bo'lgan. MVP Taligentning C++ asosidagi CommonPoint muhitida ilovalarni ishlab chiqish uchun asosiy dasturlash modelidir. Keyinchalik namuna Taligent tomonidan Javaga ko'chirildi.

1998 yilda Taligent faoliyatini to'xtatgandan so'ng, Dolphin Smalltalk kompaniyasidan Endi Bower va Bler MakGlashan MVP modelini Smalltalk foydalanuvchi interfeysi ramkasi uchun asos yaratish uchun moslashtirdilar. 2006 yilda Microsoft MVP ni o'z hujjatlariga va .NET Framework foydalanuvchi interfeysi dasturlash misollariga qo'shishni boshladi.

MVP (Model — View- Presenter) arxitektura patterni bizga MVC ning o’rnini bosuvchi muqobil variant sifatida ko’rilishi mumkin. Dastur arxitekturasi sifatida MVC dan foydalanish bir qancha qiyinchiliklarni keltirib chiqarishi mumkin:

- Biznes logikaning aksariyati qismi Controllerda joylashadi. Ilovani ishlash muddati davomida bu fayl kattalashib boradi va kodni saqlash bir muncha qiyinlashadi.

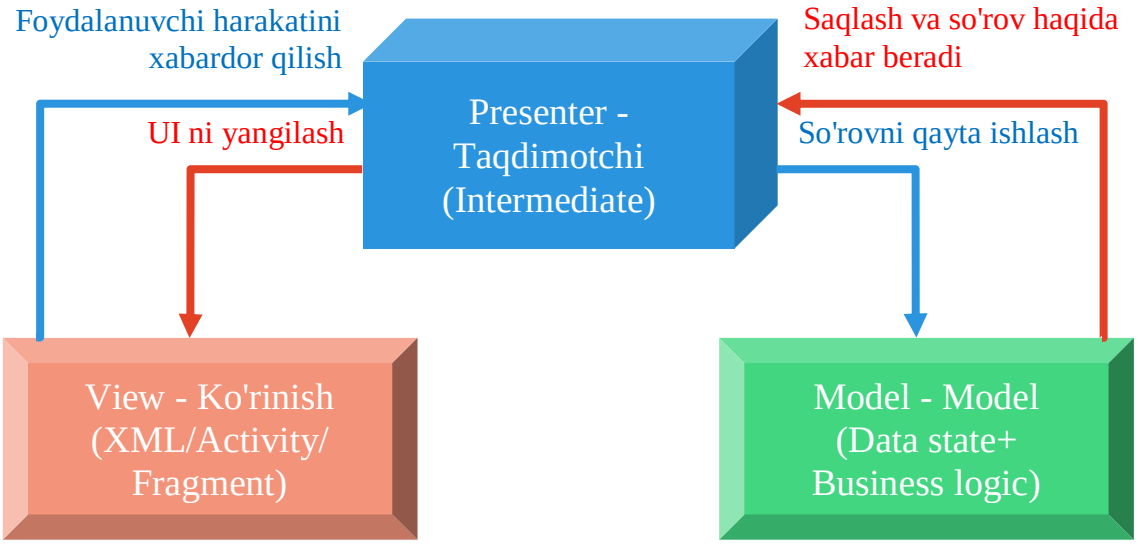
- Controller va Viewni ikkalovi ham bir Activity/Fragment da bo'lsa, UI va ma'lumotlarni almashish mexanizmlari ham bir-biri bilan birlashib ketadi. Bu keyinchalik dastur xususiyatlariga o'zgartirish kiritishda muammo tug'diradi.
- Turli darajadagi Unit Testing jarayonlarini o'tkazish murakkab bo'ladi, chunki testdan o'tayotgan qismning asosi Android SDK komponentlariga muhtoj.

MVP patter ni MVC ning bu muammolarini yechib beradi va proyekt uchun yoziladigan kodlarni yaxshi strukturaga olib keladi [3]. MVP keng qabul qilinishining sababi shundaki, u modullikni, test qilish imkonini va kod bazangizni yanada tushunarli ko'rinishda saqlanishini ta'minlaydi. Arxitekturalarni qatlamlarga bo'lish dasturni ishlab chiqish vaqti uchun juda muhimdir. U quyidagi uchta komponentdan iborat (1-rasm):

Model: ma'lumotlarni saqlash uchun ajratilgan qism. U domen logikasi va ma'lumotlar bazasi va tarmoq qatlamlari bilan aloqa qilish uchun javobgardir.

View: UI (foydalanuvchi interfeysi). U Presenterni xabardor qilish uchun ma'lumotlarning vizualizatsiyasini ta'minlaydi va foydalanuvchining harakatlarini kuzatib boradi.

Presenter: U modeldagi ma'lumotlarni oladi va ekranga nimani chiqarishni aniqlash uchun UI logikasini qo'llaydi. U View holatini boshqaradi va foydalanuvchining Viewdan kiritilgan ma'lumotga muvofiq harakat qiladi.



1-rasm: MVP (Model — View- Presenter) arxitektura patterni

MVP arxitekturasining muhim jihatlariga quyidagilarni kiritishimiz mumkin.

1. View-Presenter va Presenter-Model o’rtasidagi aloqa interface orqali amalga oshiriladi.

2. Bir vaqtni o’zida bir Presenter faqat bitta View ni boshqara oladi. Presenter va View o’rtasida one-to-one aloqasi qo’llaniladi.

3. Model va View klass bir birining mavjud ekanligi haqida hech narsani bilmaydilar.

MVP arxitekturasining afzalliklari

- View and Presenter dan qayta foydalanish mumkin
- Kod saqlash uchun qulay va o’qish uchun tushunarli
- Biznes Logika UI dan alohida bo’lgani uchun test qilish oson

MVP arxitekturasining kamchiliklari

- View va Presenter o’rtasida zich bog’lanish
- Layerlar o’rtasidagi o’zaro muloqot uchun juda ko’p interface
- Kod hajmi juda katta

Xulosa qilib aytadigan bo’lsak MVP arxitekturasi dasturlashda qo'llaniladigan strategiyalardan biridir. Bu strategiya, Model, View va Presenter nomli qismlardan iborat. Model, ma'lumotlar bilan ishlash va ularni saqlashda ishtirok etadi. View, foydalanuvchiga ma'lumotni ko'rsatadi va interfeys o'zgarishlarini boshqaradi. Presenter esa View va Model orasidagi aloqalarni boshqaradi [4].

MVP arxitekturasi loyihalar tuzilishini aniq va oson qiladi, kodni test qilish va qo'llanishni osonlashtiradi. Ushbu strategiya yangi funksiyalar qo'shilishi va mavjud funksiyalar o'zgartirilishi uchun ham qulay bo'ladi.

FOYDALANILGAN ADABIYOTLAR RO‘YXATI

1. Махкамов, Ш. (2023). Теоретические основы базы данных (мб) и системы управления базами данных (мббт). Информатика и инженерные технологии, 1(1), 90-94.

2. Mahkamov Sh. Davlat boshqaruv tizimida innovatsion texnologiyalarni qo‘llash bosqichlari / Mahkamov Shohruh Sarvar o’g’li - International scientific-practical conference - 2023. № 1, - 275-277 b.

3. Аликулов, С. Т., & Махкамов, Ш. С. (2018). ДИДАКТИЧЕСКАЯ КОМПЕТЕНТНОСТЬ ПЕДАГОГА. In Развитие интеллектуально-творческого потенциала молодежи: из прошлого в современность (pp. 150-152).

4. Махкамов, Ш. (2024). Davlat boshqaruv tizimining vujudga kelishi hamda davlat va uning moxiyati. Новый Узбекистан: наука, образование и инновации, 1(1), 379-382.

5. Махкамов, Ш. С. (2024). ИННОВАЦИОННЫХ ТЕХНОЛОГИЙ В СИСТЕМУ ГОСУДАРСТВЕННОГО УПРАВЛЕНИЯ КАК ФАКТОР ЕЁ РАЗВИТИЯ И СОВЕРШЕНСТВОВАНИЯ.

6. Toxir, A., & Lobar, A. (2023). Mobil ilovalar orqali yosh bolalarda uchraydigan nutq buzilishlarini bartaraf etish. In Uz-Conferences (Vol. 1, No. 1, pp. 906-911).

7. Toxir Turg'un o'gli, A. (2023). o‘rinlarni almashtirish usullari. In Uz-Conferences (Vol. 1, No. 1, pp. 133-138).

8. Norqo’ziyev , Q. (2023). Mobil robotlar uchun yo’lni rejalashtirish algoritmi. Research and Implementation. извлечено от https://fer-teach.uz/index.php/rai/article/view/746

9. Тоджиев, А., & Норкузиев, К. (2023). The role of artificial intelligence technology in individualized teaching . Информатика и инженерные технологии, 1(2), 153–156. извлечено от https://inlibrary.uz/index.php/computer-engineering/article/view/25014

10. Абдумаликов А. А. и др. Модель и алгоритмы процесса устройств контроля и мониторинга управления энергоснабжением //Journal of innovations in scientific and educational research. – 2023. – Т. 6. – №. 2. – С. 120-129.