KAPITOLA 11

Pokročilá témata

V této kapitole:

- Získáte přehled o architektuře operačního systému iOS
- Stručně představíme dva nejpoužívanější frameworky
- Naučíte se, jak se nastavují záležitosti související s ochranou soukromí uživatele
- Ukážeme si možnosti nastavování parametrů pro aplikaci na úrovni operačního systému
- V komplexním příkladu si ukážeme možnosti využití strojového učení a umělé inteligence
- Naučíte se zakomponovat do aplikací rozšířenou realitu

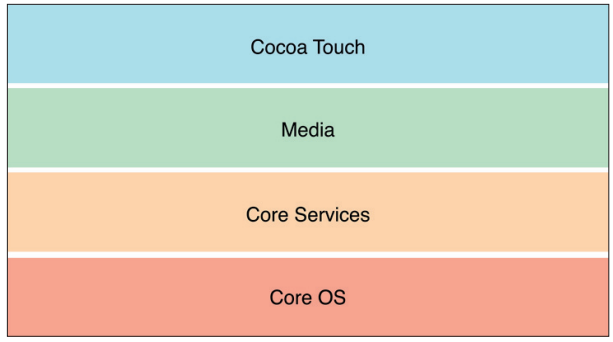
Začátečník, který vytváří jednodušší aplikace a na nich se průběžně učí, zpočátku nepotřebuje vědět, co je na pozadí jeho aplikace. Proto jsme téma architektury iOS zařadili až do této kapitoly. Dříve než začnete vyvíjet složitější projekty, je potřeba poznat aspoň základní principy vývoje pro danou platformu.

Architektura operačního systému iOS

Operační systém iOS vznikl jako odnož macOS pro přenosná zařízení. Je postaven na bázi systému Darwin s jádrem XNU. Architektura iOS je rozdělena do čtyř vrstev. Skládá se z jádra **Core OS**, základních služeb **Core Services**, multimediální vrstvy **Media** a **Cocoa Touch,** která se využívá při tvorbě uživatelského rozhraní**.** Na nejvyšší úrovni architektury jsou **aplikace.**

Při vývoji aplikací se, pokud je to možné, doporučuje preferovat frameworky vyšších vrstev, jelikož vyšší vrstvy zjednodušují programování tím, že poskytují vyšší úroveň objektově orientované abstrakce. Tím redukují množství kódu, které je potřeba naprogramovat, zrychlují vývoj a aplikace je spolehlivější a bezpečnější díky přehlednému kódu, v němž se dá při ladění dobře orientovat. Samozřejmě když požadovaná funkce není k dispozici na vyšších vrstvách, musí vývojář použít funkce frameworků z nižší vrstvy.

Popis začneme od nejvyšší vrstvy, která je situovaná nejdále nad hardwarem, ale nejblíže k aplikaci.



Obrázek 11.1: Základní blokové schéma architektury iOS

Cocoa Touch

Tato vrstva je pro vývojáře velmi důležitá, jelikož obsahuje klíčové frameworky zapouzdřující stavební kameny uživatelského rozhraní, tedy třídy, které mají vizuální reprezentaci jako ovládací prvky a dokážou provádět určité akce. Cocoa Touch obsahuje frameworky, jež se nejčastěji používají při vývoji aplikací. Tato vrstva je kompletně napsaná v Objective-C, je založena na standardním systému macOS Cocoa API, který byl upraven tak, aby vyhovoval potřebám mobilních platforem Apple iPad/iPhone/iPod Touch s dotykovým displejem. Cocoa Touch obsahuje podporu klíčových technologií, jako například multitasking, notifikace, „dotykové" uživatelské rozhraní a hodně dalších systémových služeb nejvyšší úrovně.

Media

Tato vrstva obsahuje technologie na práci s multimédii, tj. s grafikou, zvukem a videem. Všechny zmíněné atributy jsou klíčové proto, aby byla aplikace atraktivní a uživatelsky příjemná. Na úrovni vrstvy Media najdete grafické knihovny Core Graphics (Quartz), OpenGL, Metal, Photos Library, Animation, knihovny na práci se zvukem, například Media player, OpenAL či Core Audio. Také podporu technologie AirPlay a podobně.

Core Services

Jak vyplývá z názvu, tato vrstva obsahuje základní systémové služby pro aplikace, například frameworky Core Foundation a Foundation, kde jsou definovány základní typy. Na úrovni této vrstvy jsou také definovaná rozhraní na používání různých technologií, jako je interakce s cloudovou službou iCloud, geolokačními a mapovými službami, webovými službami, sociálními sítěmi, interní databází SQLite, vrstvou na telefonování, posílání SMS a podobně.

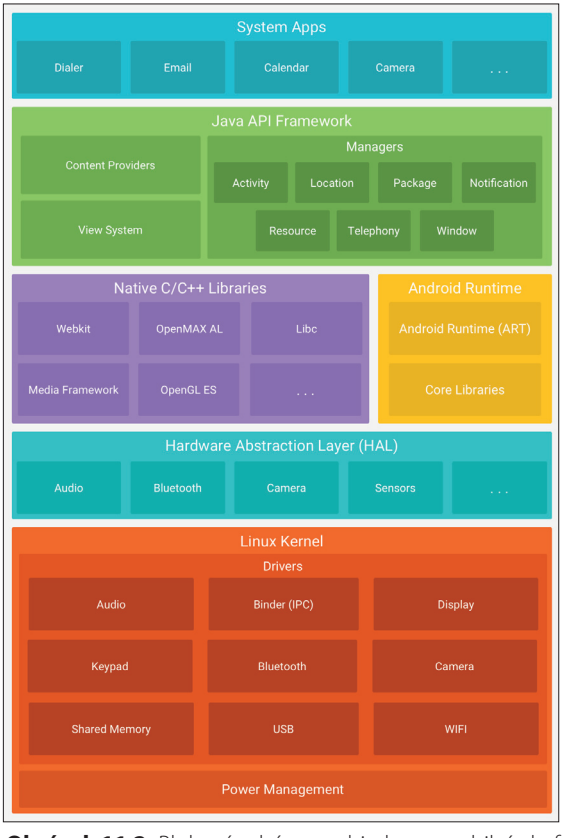
Core OS

Vrstva Core OS obsahuje nízkoúrovňové funkce určené pro vyšší vrstvy. S největší pravděpodobností funkce této vrstvy při vývoji aplikaci využívat nebudete. Výjimkou jsou různé aplikace, které řeší bezpečnost nebo komunikaci s externím hardwarem.

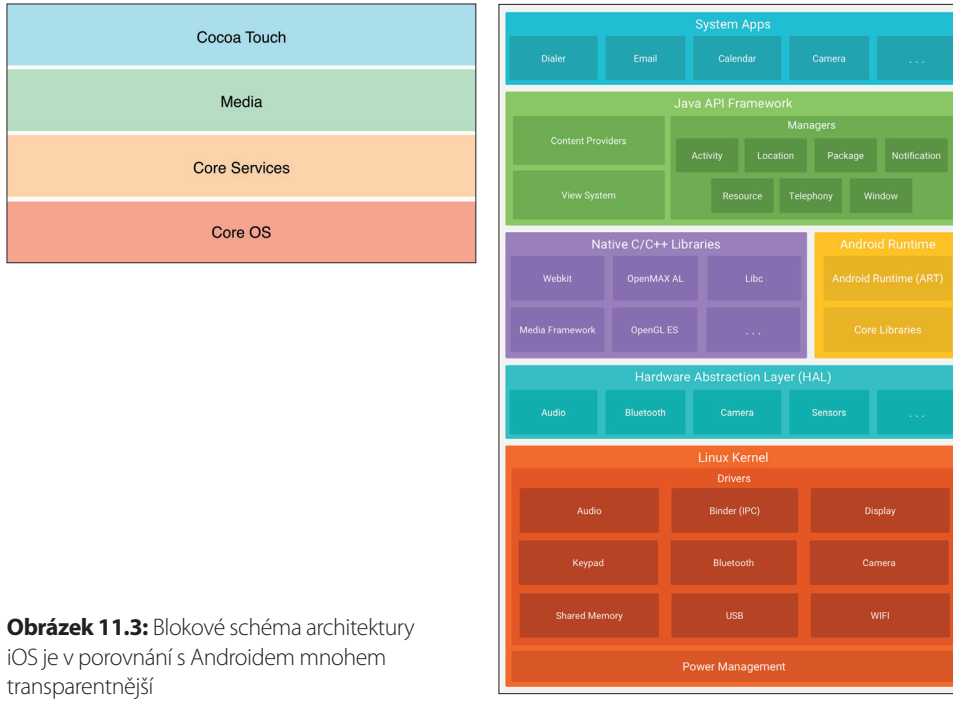
Při tvorbě aplikací by měli vývojáři využívat aplikační frameworky na co nejvyšší úrovni, kde je možné využívat objektově orientované abstrakce, což výrazně zrychluje a zefektivňuje vývoj. Přestože jsou tyto vrstvy abstraktní, jejich primárním úkolem není maskovat technologie na nižších úrovních, jsou také k dispozici pro vývojáře, kteří upřednostňují vývoj na nativní úrovni například kvůli rychlosti, případně je použít musí, protože funkcionalita, kterou potřebují do aplikace implementovat, není na vyšších vrstvách zpřístupněna. Když to sumarizujeme, v projektové fázi byste měli prozkoumat, zda technologie na úrovni Cocoa Touch splňují vaše požadavky, a prioritně je využívat.

Pro migranty z Androidu

Situace, kdy nastane požadavek na portování úspěšné mobilní aplikace na jinou platformu, je poměrně častá, takže častá bude i situace, kdy vývojáři aplikace pro Android budou postaveni před nelehký úkol portovat aplikaci, kterou vytvořili pro Android, i na mobilní platformu iOS. Stačí letmý pohled na obě architektury a zjistíte, že v iOS jsou vrstvy ploch mnohem přehledněji hierarchicky uspořádané a všechny vrstvy jsou účelově přístupné.



Obrázek 11.2: Blokové schéma architektury mobilní platformy Android



Obrázek 11.3: Blokové schéma architektury iOS je v porovnání s Androidem mnohem transparentnější

Nebudeme řešit podrobně všechny atributy vývoje pro iOS, který je pro vývojáře pro Android novou platformou, o tom je přece celá tato publikace, jen stručně vyjmenujeme iOS ekvivalenty základních stavebních pilířů aplikace pro Android.

- Activity → ViewController
- Intenty → Segue, ViewController
- Service → „Background Mode“ + specifické volání API
- Content Provider → CoreData
- Layout → Storyboard a Scene

Frameworky

Stručně si představíme dva nejpoužívanější frameworky.

Foundation

Tento framework je základní knihovnou při vývoji aplikací v jazyce Objective-C. Třídy tohoto frameworku snadno poznáte podle prefixu NS. Prefix NS v názvech některých objektů znamená, že tento objekt je odvozen od `NSObject`. NS znamená NEXTSTEP, což je technologie vyvinutá společností NeXT Software, kterou Apple koupil v roce 1997. Tato technologie se později stala základním pilířem macOS. Základní třídy frameworku Foundation:

- `NSObject` – bázová třída. Všechny objekty v Objective-C jsou odvozeny od třídy `NSObject`.
- `NSString` – třída na manipulaci s datovými typy, do kterých se ukládají textové řetězce.
- `NSValue` – třída funguje jako kontejner na objekty určené na ukládání údajů.
- `NSNumber` – třída je odvozena od bázové třídy `NSValue`. Je určená k ukládání numerických datových typů `Interger`, `Double` a `Float`.
- `NSArray` – třída je určena na práci s poli objektů.

UIKit

Zatímco je framework Foundation zaměřen více na aplikační logiku, UIKit řeší práci s objekty uživatelského rozhraní. Základní třídy:

- `UIApplication` – třída řeší řízení aplikace, vývojáři aplikací ale s touto třídou málokdy potřebují pracovat.
- `UIViewController` – instance této třídy jsou jednotlivé pohledy (obrazovky) uživatelského rozhraní aplikace. Na úrovni této třídy se definují layouty a propojení více pohledů.
- `UIView` – každý kontrolér musí obsahovat komponentu `UIView`, která zapouzdřuje prvky uživatelského rozhraní.
- `UIGestureRecognizer` – základní metodou interakce zařízení s iOS s uživatelem jsou dotyky a gesta. Abstraktní třída `UIGestureRecognizer` řeší zpracování dotyků a gest.

Nastavení ochrany soukromí uživatele

Některé hardwarové komponenty a s nimi spojené funkce, především mikrofon na nahrávání zvuku, kamera na snímání fotografií a videa, lokalizační a další služby a také údaje z kalendářů nebo seznamu kontaktů mohou prozradit o uživateli věci, které by si určitě nepřál zveřejňovat. Proto aplikace pro iOS vyžadují na takovéto akce povolení uživatele. Tato povolení se deklarují v souboru *Info.plist*.

Aplikace musí požádat o přístup k citlivým údajům uživatele nebo zařízení, která jsou chráněna nastavením autorizace systému iOS v čase, když aplikace potřebuje údaje. Musíte uvést účel, k čemu aplikace tyto údaje potřebuje. Údaje chráněné nastaveními autorizace systému iOS zahrnují umístění, kontakty, události kalendáře, připomínky, fotografie, média a mnoho dalších typů údajů, které mohou potenciálně zasáhnout do soukromí uživatele.

Buďte transparentní a poskytněte uživatelům informaci o tom, jak se budou jejich údaje používat. Když odešlete svou aplikaci do App Store, zadejte URL adresu stránky, kde deklarujete svou politiku ochrany osobních údajů. Popis této politiky můžete také zahrnout do popisu aplikace. Umožněte uživateli kontrolu nad citlivými údaji. Navrhněte nastavení tak, aby uživatel mohl podle potřeby zakázat přístup k určitým typům citlivých informací.

V aplikaci požádejte a použijte minimální možné množství údajů uživatele nebo zařízení nezbytně potřebných ke splnění daného úkolu. Nepožadujte přístup ani shromažďování údajů jen proto, že si myslíte, že by to mohlo být užitečné později. Učiňte také přiměřené kroky na

ochranu údajů uživatelů a zařízení, které shromažďujete ve svých aplikacích. Při lokálním ukládání takovýchto informací údaje ochraňte například šifrováním.

Nastavení žádosti o souhlas uživatele vyžadují frameworky: Calendar, Contact, Reminder, Photo, Bluetooth Sharing, Microphone, Camera, Location, Heath, HomeKit, Media Library, Motion, CallKit, Speech Recognition, SiriKit, TV Provider.

Pro tyto frameworky se nastavují klíče.

- Calendar: Privacy – Calendars Usage Description
- Reminder: Privacy – Reminders Usage Description
- Contact: Privacy – Contacts Usage Description
- Photo: Privacy – Photo Library Usage Description
- Value: $(PRODUCT_NAME) photo use
- Bluetooth Sharing: Privacy – Bluetooth Peripheral Usage Description
- Microphone: Privacy - Microphone Usage Description
- Camera: Privacy – Camera Usage Description
- Location: Privacy – Location Always Usage Description
- Location: Privacy – Location When In Use Usage Description
- Heath: Privacy – Health Share Usage Description
- Health: Privacy – Health Update Usage Description
- HomeKit: Privacy – HomeKit Usage Description
- Media Library: Privacy – Media Library Usage Description
- Motion: Privacy – Motion Usage Description
- Speech Recognition: Privacy – Speech Recognition Usage Description
- SiriKit: Privacy – Siri Usage Description
- TV Provider: Privacy – TV Provider Usage Description

Nastavení parametrů aplikace

Aplikace může zobrazit nastavení a umožnit jejich změnu přímo ve svém uživatelském rozhraní nebo v nastavení zařízení. Ve svém uživatelském rozhraní může aplikace k tomuto účelu vyhradit samostatnou obrazovku (**View**), samostatnou kartu a podobně. Nastavení zařízení jsou soustředěna v aplikaci *Nastavení* (*Settings*), kde kromě parametrů operačního systému můžete nastavovat i parametry aplikací, které tento způsob podporují.

Nastavování parametrů se realizuje ve většině aplikací a vývojáři by potom téměř v každé aplikaci museli programovat pro tento účel uživatelské rozhraní. Pro nastavování parametrů v nastavení zařízení není potřeba vytvářet ve vašich aplikacích uživatelské rozhraní, protože aplikace *Nastavení* poskytuje standardní rozhraní nejen na testování parametrů zařízení a jeho operačního systému, ale i na nastavování parametrů aplikací. Přispívá to ke konzistenci uživatelské zkušenosti.