React Native - APP Size

1. Tamanho da aplicação

Hoje em dia, uma aplicação não pode simplesmente só funcionar bem, ela tem que pensar em questões de usabilidade, estética e sim... Tamanho de download.

Não esperamos que tenhamos tanta informação "chumbada" no código, o famoso *hardcoded*, espera-se que nossa aplicação final fique com um tamanho reduzido.

Na loja da Google, já não se aceita mais APKs como formato para que suba novas aplicações, hoje o que se espera é o AAB que já reduz drasticamente o tamanho do pacote final, mas como didática, vou demonstrar os tamanhos extraindo uma APK e comparando.

2. Configuração iOS - On-Demand Resources

Os recursos sob demanda são conteúdos de aplicativos hospedados na App Store e separados do pacote de aplicativos relacionado que você baixou. Eles permitem pacotes de aplicativos menores, downloads mais rápidos e conteúdo de aplicativo mais rico. O aplicativo solicita conjuntos de recursos sob demanda e o sistema operacional gerencia o download e o armazenamento. O app usa os recursos e depois libera a requisição. Após o download, os recursos podem permanecer no dispositivo por vários ciclos de inicialização, tornando o acesso ainda mais rápido. Docs

Por padrão, as novas versões do React Native vem com este recurso ativado. Verifique se o seu está. Se não, selecione o `target -> Build settings -> Assets -> Enable On-Demand Resources`

3. Configuração Android

Para tornar seu aplicativo o menor possível, você deve habilitar a redução em seu build de lançamento para remover código e recursos não utilizados. Ao habilitar a redução, você também se beneficia da ofuscação , que reduz os nomes das classes e membros do seu aplicativo, e da otimização , que aplica

estratégias mais agressivas para reduzir ainda mais o tamanho do seu aplicativo. https://developer.android.com/build/shrink-code#groovy

3.0. Android API Level

Manter o targetSdk atualizado é crucial para aplicativos móveis Android. Atualizar o targetSdk para a versão mais recente garante que o aplicativo pode aproveitar os recursos e melhorias mais recentes oferecidos pela nova versão do Android, melhorando a experiência do usuário. Além disso, o Google Play só aceita novos aplicativos e atualizações de aplicativos que tenham um targetSdk atualizado. Manter este valor atualizado garante a compatibilidade do aplicativo com os requisitos de lançamento do Google Play, permitindo que os desenvolvedores alcancem a maior audiência possível.

3.1. Shrink, obfuscate, e otimização

Acessaremos `android/app/build.gradle` e habilitaremos uma opção: `enableProguardInReleaseBuilds`.

`enableProguardInReleaseBuilds` irá ativar o uso do ProGuard, que ofuscará nosso código e nas novas versões do React Native também ativará o `minify`, responsável por tentar diminuir o tamanho dos nossos bundles finais.

```
def enableProguardInReleaseBuilds = true

android {
    // Resto do código...
    buildTypes {
        debug {
            signingConfig signingConfigs.debug
        }
        release {
            signingConfig signingConfigs.debug
            minifyEnabled enableProguardInReleaseBuilds
            shrinkResources enableProguardInReleaseBuilds
            proguardFiles getDefaultProguardFile("proguard-
android.txt"), "proguard-rules.pro"
        }
    }
}
```

Para realizar o build, primeiro iremos rodar o seguinte comando: `cd android && ./gradlew clean` para limparmos o projeto Android e garantirmos um build limpo.

3.2.Comparação de APKs (Android Studio - APK Analyzer)

As configurações `minify`, `proguard` e `shrink` ajudam a manter o tamanho do aplicativo pequeno para download. Além disso, habilitar `minify`, `proguard` e `shrink` ajuda a reduzir ainda mais o tamanho do aplicativo. `Minify` remove o código não utilizado, `proguard` ofusca o código tornando-o menor e mais difícil de ser copiado, e `shrink` remove recursos não utilizados.