Naar de stores

MijnBesluit · Google Play en Apple App Store

Beide trajecten hebben één ding gemeen aan het begin: een **D-U-N-S-nummer** voor de rechtspersoon (de B.V.). Dat kan weken duren en is nodig voor zowel Google als Apple — vraag het dus één keer aan en laat het traject alvast lopen terwijl je verder werkt.

Google Play (Android)

1	**D-U-N-S-nummer aanvragen** Voor het organisatie-account bij Google Play. Zie de opmerking hierboven — dit is dezelfde aanvraag die je ook voor Apple nodig hebt.
2	**Organisatie-account bij Google Play aanmaken** Met dat D-U-N-S-nummer erbij.
3	**De bundel bouwen** `tools\Apk-maken.ps1 -Aab` — maakt de bundel die de store wil, in plaats van de losse APK die je nu voor eigen gebruik gebruikt.
4	**Keystore uploaden** Hetzelfde sleutelbestand waarmee ook je eigen testversies worden ondertekend.
5	**Winkelvermelding invullen** Beschrijving, screenshots, categorie, contactgegevens.
6	**Formulier “Databeveiliging” invullen** Welke gegevens de app verzamelt en hoe daarmee wordt omgegaan.

Bewaar een back-up van `D:\MijnBesluitSleutels\mijnbesluit-app.keystore`**.** Zonder dat bestand kan er nooit meer een update voor dezelfde app in de store komen — Google accepteert dan geen nieuwe versie meer onder hetzelfde app-record.

Apple App Store (iOS)

De app zelf is dezelfde als de Android-versie: `MijnBesluit.AppUi`, `MijnBesluit.Client` en het socketprotocol zijn gedeeld. Op iOS draait dezelfde HTML in een WKWebView. `net10.0-ios` staat al in het project, maar staat standaard uit — aanzetten met `-p:MetIos=true`.

1	**D-U-N-S-nummer aanvragen voor de B.V.** Zie hierboven — deze aanvraag telt voor beide stores.
2	**Apple Developer Program, als organisatie** $99 per jaar. Nodig: een echte rechtspersoon met dat D-U-N-S-nummer, een e-mailadres op je eigen domein, en een werkende website op dat domein.
3	**Een Mac regelen, Xcode erop, “Pair to Mac” aan de praat** Apple's bouwgereedschap draait alleen op macOS. Visual Studio op Windows verbindt er over het netwerk mee (SSH) en laat de Mac compileren en ondertekenen. Op de Mac moet Xcode staan, één keer met de hand geopend zijn, en Remote Login aan.
4	**Bouwen en op een echt toestel zetten** Geen simulator: de socketverbinding op poort 5200 en het toestelslot (Face ID/Touch ID) wil je op echte hardware zien, ook over 4G en niet alleen over wifi.
5	**Uitproberen** Inloggen met een code, de socketverbinding, de kluis, het toestelslot met Face ID.
6	**Meldingen via APNs instellen** Een APNs-sleutel, de Push Notifications-capability, een provisioning profile en een Entitlements.plist. Zonder dit geeft “Toestelmeldingen” op iOS nu gewoon niets terug; de rest van de app werkt dan al wel.
7	**Inzenden ter beoordeling** Zie de vier punten hieronder die Apple daarbij expliciet vraagt.

Wat Apple bij het inzenden vraagt

• **Account verwijderen in de app** — verplicht, staat er al.
• **App Privacy** — hetzelfde soort formulier als Databeveiliging bij Google; de privacyverklaring staat in `Privacytekst.razor`.
• **Exportverklaring** (`ITSAppUsesNonExemptEncryption`) — deze app versleutelt een kluis, dus “nee” is hier niet vanzelfsprekend zoals bij een app die alleen HTTPS gebruikt.
• **Sign in with Apple** is niet verplicht — dat geldt alleen bij inloggen via Google of Facebook; MijnBesluit gebruikt e-mail met een code.

Twee vallen waar je vroeg in loopt: (1) een ontbrekende gebruikstoelichting in `Info.plist` (Face ID, camera, microfoon, fotobibliotheek) laat iOS de app stilletjes afsluiten, zonder foutmelding — ze staan er nu allemaal in, dus check dit opnieuw zodra er iets nieuws bij komt dat het toestel moet vrijgeven. (2) de socket op poort 5200 heeft geen ATS-uitzondering nodig (eigen TLS-verbinding, geen URL-verkeer) — een uitzondering die je niet nodig hebt levert bij de beoordeling alleen vragen op.

Bronnen: `docs/iOS-app.md` en de eerdere afspraken over `tools\Apk-maken.ps1` en het sleutelbeheer voor Google Play.