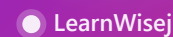


Firma, distribuzione, aggiornamenti e il capstone FieldOps

Guida della lezione · Guida al lab / esame



Guida della lezione

Firma del codice, invio agli store, distribuzione enterprise, versionamento, strategie di aggiornamento tra gli shell, telemetria e la consegna del capstone. Che cosa significa per un team che porta un'app Wisej.NET oltre la scheda del browser — e come affrontare questo modulo.



Focus di produzione

Firma del codice, invio agli store, distribuzione enterprise, versionamento, strategie di aggiornamento tra gli shell, telemetria e la consegna del capstone.

1. Il problema in produzione

FieldOps termina questo corso come un server e quattro modi per raggiungerlo: una scheda del browser, una PWA installata, uno shell desktop per Windows e uno shell mobile su Android e iOS. I tre confezionati sono file che un sistema operativo si rifiuta di installare se non li ha firmati un'identità di cui si fida, e ciascuno viaggia su un canale diverso con la propria coda di revisione e la propria velocità di rollout. Puoi ridistribuire il server prima di pranzo; la build iOS che un tecnico ha in mano può essere indietro di tre settimane, e deve comunque funzionare.

2. Concetti chiave

Concetto	Perché è importante
firma Authenticode	La firma che un certificato di firma del codice applica a un installer Windows o a un pacchetto MSIX, dimostrando che il file non è stato alterato dopo la firma e indicando un editore verificato.
timestamp di controfirma	Un timestamp attendibile registrato al momento della firma, così che l'installer continui a risultare valido anche dopo la scadenza del certificato di firma stesso.
keystore Android e chiave di upload	La chiave che firma una build Android; con Play App Signing firmi con una chiave di upload e Google firma di nuovo con la chiave di firma dell'app che conserva per quel pacchetto.
certificato di distribuzione Apple e provisioning profile	Il certificato identifica il tuo team; il profilo lega un App ID, quel certificato, gli entitlement e, fuori dall'App Store, i dispositivi consentiti.
distribuzione enterprise	Consegnare una build privata ai dispositivi di un'unica organizzazione — managed Google Play, app personalizzate di Apple Business Manager, un pacchetto MSIX distribuito da Intune — invece di pubblicarla pubblicamente.
versione minima dello shell	La build di shell più vecchia, per piattaforma, che il server attuale supporta ancora; qualsiasi versione inferiore viene fermata alla connessione prima di raggiungere qualsiasi dato.
evento di utilizzo di una funzionalità del dispositivo	Un record scritto ogni volta che viene eseguita una capacità dello shell — fotocamera, firma, posizione, push, invio della coda offline — con l'esito più la piattaforma e la versione dello shell.

3. Il filo della lezione

Che cosa dimostra una firma, e le tre di cui FieldOps ha bisogno

Una firma del codice fa esattamente due affermazioni: i byte sono quelli che sono stati firmati, e li ha firmati un'identità con un nome. Non dice nulla sul fatto che il codice si comporti bene, e l'intera catena di distribuzione poggia su quelle due affermazioni e nient'altro. Firma il pacchetto Windows con un certificato di firma del codice rilasciato alla tua azienda e aggiungi sempre un timestamp attendibile; senza, l'installer smette di risultare valido il giorno in cui il certificato scade. Se rilasci lo shell desktop come MSIX, il soggetto del certificato deve corrispondere al nome dell'editore nel manifest del pacchetto, altrimenti Windows lo rifiuta senza appello.

Android usa un modello diverso: la build viene firmata con una chiave conservata in un keystore, e l'identità che interessa al dispositivo è quella chiave, non un'autorità di certificazione. Iscriviti FieldOps a Play App Signing, dove firmi ogni release con una chiave di upload e Google la verifica, la rimuove e firma di nuovo con la chiave di firma dell'app che conserva. Questo elimina il guasto peggiore della distribuzione mobile, una chiave persa che rende l'app per sempre impossibile da aggiornare, perché l'unica chiave che puoi perdere è quella di upload e può essere reimpostata.

Apple richiede due artefatti. Il certificato di distribuzione identifica il tuo team; il provisioning profile dice che questo App ID, firmato da quel certificato, può usare questi entitlement e — fuori dall'App Store — può girare su questi dispositivi. Le notifiche push, che FieldOps usa per annunciare un nuovo ordine di lavoro, sono un entitlement, quindi un profilo precedente a quella capacità produce una build che si installa e poi, in silenzio, non riceve nulla. Il capstone chiede tutti e tre gli artefatti firmati più una nota per piattaforma che registra chi custodisce la chiave, dove si trova e quando scade.

Store, enterprise o diretta: una decisione per piattaforma

Esistono tre canali e ogni piattaforma ne sceglie uno. Una scheda pubblica sullo store è adatta a un'app che chiunque può installare: Google Play e l'App Store ti danno hosting, aggiornamenti differenziali, rollout graduali e una coda di revisione che si misura in ore o giorni. La distribuzione enterprise si adatta meglio a FieldOps, perché i tecnici sono dipendenti con dispositivi gestiti: managed Google Play pubblica in privato per un'unica organizzazione e assegna l'app tramite il loro EMM, Apple Business Manager distribuisce un'app personalizzata a organizzazioni con un nome che la installano tramite MDM, e su Windows il pacchetto MSIX firmato viene distribuito tramite Intune. Il download diretto — il tuo server che ospita l'installer firmato e un feed delle versioni — resta utile per lo shell desktop, ed è di fatto inutilizzabile su iOS.

Il browser e la PWA fanno eccezione, ed è proprio il senso dell'intero corso: nessun canale, nessun certificato, nessuna revisione. Fai il deploy del server e il successivo caricamento di pagina è la nuova versione. Ogni ora spesa per mantenere FieldOps pienamente utilizzabile in un semplice browser è un'assicurazione contro la coda di uno store.

Preparare un invio è lavoro vero, quindi il lab chiede una bozza della scheda dello store anche dove distribuisce in privato: nome e descrizione scritti per un tecnico e non per un acquirente, screenshot in ogni dimensione di dispositivo richiesta, una dichiarazione sulla privacy e sulla sicurezza dei dati che elenca onestamente i dati di fotocamera, foto e posizione che FieldOps raccoglie, e un contatto di supporto. La configurazione enterprise è l'immagine speculare: gruppi di assegnazione, configurazione gestita come l'indirizzo del server, e chi nel team IT del cliente approva una nuova versione.

Il server corre, gli shell arrancano

Poiché gli shell non riescono a tenere il passo del server, la regola di compatibilità deve essere esplicita. Dai al server una propria versione e una versione minima dello shell per piattaforma — la build più vecchia con cui è ancora disposto a parlare. Lo shell invia piattaforma e versione alla connessione, come parametri sull'indirizzo che carica o tramite l'handshake del bridge, e il server le registra entrambe in `Application.Session`, così che ogni riga di log e ogni evento di telemetria successivi possano nominarle. Sotto la versione minima, il tecnico riceve una schermata di aggiornamento bloccante che punta alla destinazione giusta per quel canale e nessun dato degli ordini di lavoro viene caricato. Alla versione minima o sopra, ma dietro l'ultima, mostra un avviso che si può chiudere e lasciarlo continuare a lavorare.

```

public ShellGate AdmitShell(string platform, Version shellVersion)
{
    var minimum = _config.MinimumShellVersion(platform);    // per esempio 2.4.0
    if (shellVersion < minimum)
        return ShellGate.MustUpdate(minimum, _config.UpdateTarget(platform));

    Application.Session["ShellPlatform"] = platform;
    Application.Session["ShellVersion"] = shellVersion;

    return shellVersion < _config.LatestShellVersion(platform)
        ? ShellGate.UpdateAvailable(_config.ServerVersion)
        : ShellGate.Current(_config.ServerVersion);
}

```

Alzare la versione minima è un atto deliberato, non un effetto collaterale di un rilascio. Alzala solo quando il server inizia a dipendere da qualcosa che gli shell più vecchi non sanno fare: un nuovo metodo del bridge, una forma del payload cambiata, un'autorizzazione che la vecchia build non ha mai richiesto. Tutto il resto, comprese le nuove schermate, deve degradare in modo accettabile sugli shell già sul campo. La disciplina vale più del controllo stesso, perché impone la domanda prima di ogni deploy — che cosa farà con questo la build Android di due mesi fa?

Telemetria che nomina lo shell, e la checklist di rilascio

Con quattro canali di consegna in produzione, una segnalazione che dice "si è bloccato" è quasi inutile. Inserisci un SDK di segnalazione dei crash in ogni shell così che i crash nativi e le eccezioni non gestite nel processo dello shell arrivino con uno stack trace, il modello del dispositivo e il numero di build, anche se sono avvenuti fuori dal tuo codice C# di UI. Sul server, registra le eccezioni non gestite allegando piattaforma e versione dello shell della sessione, così che un incidente si possa leggere da entrambe le estremità. Poi aggiungi un evento di utilizzo per ogni funzionalità del dispositivo: foto acquisita, firma salvata, posizione risolta, push ricevuto, upload in coda inviato, ciascuno con il suo esito e lo shell da cui proviene. Due settimane di questi eventi rispondono a domande che le discussioni non risolvono, per esempio se il ripiegio della fotocamera scatta perché un'autorizzazione è stata negata o perché lo shell è troppo vecchio per avere quella capacità.

La telemetria è un insieme di dati sul lavoro delle persone, quindi dichiarala. Nessun contenuto delle foto, nessuna immagine delle firme, nessun nome di cliente, nessuna coordinata grezza dove basta un identificativo dell'ordine di lavoro — e qualunque cosa tu raccolga appartiene alla dichiarazione sulla privacy che hai presentato. Il capstone si chiude poi con una checklist di rilascio, perché quattro canali di consegna hanno quattro modalità di guasto diverse e la checklist è ciò che rende un rilascio ripetibile da qualcuno che non era presente quando è stato progettato.

- Browser: server distribuito, endpoint della versione aggiornato, smoke test su una sessione a freddo.
- PWA: versione della cache del service worker incrementata, flusso di aggiornamento verificato a partire dalla build precedente.
- Desktop: installer firmato e con timestamp, firma verificata su una macchina pulita, pacchetto Intune sostituito.

- Mobile: chiavi e profili corretti, versione minima dello shell rivista, rollout graduale impostato, dashboard dei crash sotto controllo il primo giorno.



La regola del packaging

L'app resta un'app web Wisej.NET servita dal tuo server; lo shell è un veicolo di distribuzione. Mantieni ogni funzionalità funzionante in un semplice browser, poi aggiungi le capacità del dispositivo come miglioramenti progressivi dietro una verifica delle capacità.



Obiettivi di apprendimento

Alla fine di questo modulo sarai in grado di:

- ✓ Firmare i pacchetti desktop e mobile e spiegare che cosa dimostra ciascun certificato.
- ✓ Scegliere per ogni piattaforma tra distribuzione tramite store, enterprise e diretta, e preparare ciascun invio.
- ✓ Progettare una strategia di versionamento e aggiornamento in cui il server possa evolvere più velocemente degli shell.
- ✓ Aggiungere telemetria di crash e utilizzo su tutti gli shell e consegnare il capstone FieldOps.

Guida al lab / esame

Che cosa costruirai nel lab pratico, l'approccio suggerito e i deliverable richiesti.



Lab pratico

Completa il capstone FieldOps: firma l'installer per Windows e i pacchetti per Android e iOS, prepara una distribuzione enterprise per gli shell mobile e una bozza della scheda dello store, definisci una regola di compatibilità tra le versioni del server e le versioni degli shell con un controllo della versione minima all'avvio, aggiungi la segnalazione dei crash e un evento di utilizzo per ogni funzionalità del dispositivo, e consegna una checklist di rilascio che copre tutti e quattro i modelli di distribuzione.

Punto di partenza

Apri la soluzione FieldOps come l'ha lasciata il Modulo 5: il server Wisej.NET con elenco, dettaglio e flusso di stato degli ordini di lavoro, il manifest e il service worker della PWA, lo shell desktop WebView2, lo shell .NET MAUI per Android e iOS, e il bridge che espone già fotocamera, firma, posizione e push dietro le verifiche delle capacità. Tutto gira attualmente da build di debug non firmate puntate a un server di sviluppo.

Approccio suggerito

- ✓ Produci per prime le tre identità di firma — un certificato di firma del codice per Windows, un keystore con una chiave di upload registrata in Play App Signing, e un certificato di distribuzione Apple con un provisioning profile che include l'entitlement per le notifiche push — e annota per ciascuna chi custodisce la chiave e quando scade.
- ✓ Crea e firma un pacchetto di release per piattaforma, poi verifica ogni firma come farà il dispositivo di destinazione: controlla la firma Authenticode e il suo timestamp su una macchina Windows pulita, e verifica che le build mobile si installino dal canale previsto e non da un cavo di debug.
- ✓ Aggiungi l'handshake delle versioni prima che venga caricato qualsiasi altra cosa: esponi dalla configurazione la versione del server e la versione minima dello shell per piattaforma, fai in modo che ogni shell comunichi piattaforma e versione alla connessione, e memorizzale entrambe nella sessione.
- ✓ Costruisci la schermata di aggiornamento che il controllo mostra a uno shell obsoleto, con la destinazione corrispondente al suo canale, e un avviso che si può chiudere per uno shell che è soltanto indietro rispetto all'ultima build.
- ✓ Integra la segnalazione dei crash in ogni shell e un evento di utilizzo in ogni chiamata al bridge, poi scrivi la checklist di rilascio percorrendo un rilascio completo su browser, PWA, desktop e mobile.

Deliverable richiesti

- ✓ Pacchetti firmati per Windows, Android e iOS
- ✓ Configurazione della distribuzione enterprise e bozza della scheda dello store

- ✓ Controllo di compatibilità tra versioni del server e dello shell
- ✓ Telemetria di crash e utilizzo
- ✓ Checklist di rilascio per tutti i modelli di distribuzione

Domande di revisione per lo studente

- ✓ Lo shell Android di un tecnico ha sei settimane e nel frattempo il server è stato rilasciato quattro volte. Quale di quelle quattro modifiche al server avrebbe dovuto alzare la versione minima dello shell, e come lo avresti saputo in quel momento?
- ✓ L'installer Windows è stato firmato ma senza timestamp, e il certificato è scaduto il mese scorso. Che cosa fanno le installazioni esistenti, che cosa fanno i nuovi download, e qual è la strada più breve per tornare a un installer attendibile?
- ✓ L'evento di utilizzo della fotocamera mostra un forte aumento del percorso di ripiego nel browser solo su iOS. Indica tre spiegazioni compatibili con i dati, e di' quale campo aggiuntivo nell'evento permetterebbe di distinguerle.



La regola del packaging

L'app resta un'app web Wisej.NET servita dal tuo server; lo shell è un veicolo di distribuzione. Mantieni ogni funzionalità funzionante in un semplice browser, poi aggiungi le capacità del dispositivo come miglioramenti progressivi dietro una verifica delle capacità.