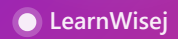


Firma, distribución, actualizaciones y el proyecto final FieldOps

Guía de la lección · Guía del laboratorio / examen



Guía de la lección

Firma de código, envío a las tiendas, distribución empresarial, control de versiones, estrategias de actualización en todos los shells, telemetría y la entrega del proyecto final. Qué significa esto para un equipo que lleva una aplicación Wisej.NET más allá de la pestaña del navegador, y cómo leer este módulo.



Enfoque de producción

Firma de código, envío a las tiendas, distribución empresarial, control de versiones, estrategias de actualización en todos los shells, telemetría y la entrega del proyecto final.

1. El problema en producción

FieldOps termina este curso como un servidor y cuatro formas de llegar a él: una pestaña del navegador, una PWA instalada, un shell de escritorio para Windows y un shell móvil en Android e iOS. Los tres empaquetados son archivos que un sistema operativo se niega a instalar a menos que los haya firmado una identidad de su confianza, y cada uno viaja por un canal distinto con su propia cola de revisión y su propia velocidad de despliegue. Puedes volver a desplegar el servidor antes de comer; la compilación de iOS que tiene un técnico en la mano puede ir tres semanas por detrás, y aun así tiene que funcionar.

2. Conceptos clave

Concepto	Por qué importa
firma Authenticode	La firma que aplica un certificado de firma de código a un instalador de Windows o a un MSIX, que demuestra que el archivo no se modificó después de la firma y nombra a un editor verificado.
marca de tiempo de la contrafirma	Una marca de tiempo de confianza registrada en el momento de la firma, para que el instalador siga validándose después de que caduque el propio certificado de firma.
keystore de Android y clave de subida	La clave que firma una compilación de Android; con Play App Signing firmas con una clave de subida y Google vuelve a firmar con la clave de firma de la aplicación que custodia para ese paquete.
certificado de distribución y perfil de aprovisionamiento de Apple	El certificado identifica a tu equipo; el perfil vincula un App ID, ese certificado, los entitlements y, fuera de la App Store, los dispositivos permitidos.
distribución empresarial	Entregar una compilación privada a los dispositivos de una organización (managed Google Play, apps personalizadas de Apple Business Manager, un MSIX distribuido por Intune) en lugar de publicarla públicamente.
versión mínima del shell	La compilación de shell más antigua, por plataforma, que el servidor actual sigue admitiendo; todo lo que esté por debajo se detiene al conectarse, antes de llegar a ningún dato.
evento de uso de una función del dispositivo	Un registro escrito cada vez que se ejecuta una capacidad del shell (cámara, firma, ubicación, notificación push, envío de datos sin conexión) con el resultado, la plataforma y la versión del shell.

3. Desarrollo de la lección

Qué demuestra una firma, y las tres que necesita FieldOps

Una firma de código hace exactamente dos afirmaciones: los bytes son los que se firmaron, y los firmó una identidad con nombre. No dice nada sobre si el código se comporta bien, y toda la cadena de distribución se apoya en esas dos afirmaciones y en nada más. Firma el paquete de Windows con un certificado de firma de código emitido a nombre de tu empresa y añade siempre una marca de tiempo de confianza; sin ella, el instalador deja de validarse el día en que caduca el certificado. Si distribuyes el shell de escritorio como MSIX, el sujeto del certificado debe coincidir con el nombre del editor del manifiesto del paquete, o Windows lo rechaza directamente.

Android sigue otro modelo: la compilación se firma con una clave guardada en un keystore, y la identidad que le importa al dispositivo es esa clave, no una autoridad de certificación. Registra FieldOps en Play App Signing: firmas cada versión con una clave de subida y Google la verifica, la retira y vuelve a firmar con la clave de firma de la aplicación que custodia. Eso elimina el peor fallo de la distribución móvil, una clave perdida que hace que la aplicación no se pueda actualizar nunca más, porque la única clave que puedes perder es la de subida, y esa se puede restablecer.

Apple necesita dos artefactos. El certificado de distribución identifica a tu equipo; el perfil de aprovisionamiento dice que este App ID, firmado por ese certificado, puede usar estos entitlements y, fuera de la App Store, puede ejecutarse en estos dispositivos. Las notificaciones push, que FieldOps usa para anunciar una nueva orden de trabajo, son un entitlement, así que un perfil anterior a esa capacidad produce una compilación que se instala y después no recibe nada, en silencio. El proyecto final pide los tres artefactos firmados más una nota por plataforma que registre quién tiene la clave, dónde está y cuándo caduca.

Tienda, canal empresarial o descarga directa: una decisión por plataforma

Existen tres canales y cada plataforma elige uno. Una ficha pública en una tienda encaja con una aplicación que cualquiera puede instalar: Google Play y la App Store te ofrecen alojamiento, actualizaciones incrementales, despliegues escalonados y una cola de revisión que se mide en horas o días. La distribución empresarial encaja mejor con FieldOps, porque los técnicos son empleados con dispositivos administrados: managed Google Play publica de forma privada para una organización y asigna mediante su EMM, Apple Business Manager distribuye una app personalizada a organizaciones concretas que la instalan con su MDM, y en Windows el MSIX firmado se distribuye mediante Intune. La descarga directa (tu propio servidor alojando el instalador firmado y un feed de versiones) sigue siendo útil para el shell de escritorio, y es prácticamente inutilizable en iOS.

El navegador y la PWA son la excepción, y esa es la idea de todo el curso: sin canal, sin certificado, sin revisión. Despliegas el servidor y la siguiente carga de página ya es la versión nueva. Cada hora dedicada a mantener FieldOps totalmente utilizable en un navegador normal es un seguro frente a la cola de una tienda.

Preparar un envío es trabajo real, así que el laboratorio pide un borrador de ficha para las tiendas incluso cuando distribuyes de forma privada: el nombre y la descripción escritos para un técnico y no para un comprador, capturas de pantalla en cada tamaño de dispositivo requerido, una declaración de privacidad y de seguridad de los datos que enumere honestamente los datos de cámara, fotos y ubicación que recoge FieldOps, y un contacto de soporte. La configuración empresarial es la imagen especular: grupos de asignación, configuración administrada como la dirección del servidor, y quién aprueba una versión nueva en el equipo de TI del cliente.

El servidor corre, los shells se arrastran

Como los shells no pueden seguir el ritmo del servidor, la regla de compatibilidad debe ser explícita. Dale al servidor su propia versión y una versión mínima del shell por plataforma: la compilación más antigua con la que todavía está dispuesto a hablar. El shell envía su plataforma y su versión al conectarse, como parámetros en la dirección que carga o mediante el protocolo inicial del puente, y el servidor guarda ambas en `Application.Session` para que cada línea de registro y cada evento de telemetría posteriores puedan indicarlo. Por debajo de la mínima, el técnico recibe una pantalla de actualización bloqueante que apunta al destino correcto para ese canal y no se carga ningún dato de órdenes de trabajo. En la mínima o por encima, pero por detrás de la última, muestra un aviso descartable y deja que siga trabajando.

```

public ShellGate AdmitShell(string platform, Version shellVersion)
{
    var minimum = _config.MinimumShellVersion(platform);    // e.g. 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);
}

```

Subir la versión mínima es un acto deliberado, no un efecto secundario de una publicación. Súbela solo cuando el servidor empiece a depender de algo que los shells antiguos no pueden hacer: un método nuevo del puente, un cambio en la forma del contenido, un permiso que la compilación antigua nunca solicitó. Todo lo demás, incluidas las pantallas nuevas, debe degradarse correctamente en los shells que ya están en uso. La disciplina vale más que la propia comprobación, porque obliga a hacerse la pregunta antes de cada despliegue: ¿qué hará con esto la compilación de Android de hace dos meses?

Telemetría que nombra el shell, y la lista de comprobación de publicación

Con cuatro distribuciones en producción, un informe de error que dice «se ha colgado» es casi inútil. Integra un SDK de informes de fallos en cada shell para que los fallos nativos y las excepciones no controladas del proceso del shell lleguen con un stack trace, el modelo del dispositivo y el número de compilación, aunque se hayan producido fuera de tu código C# de la interfaz. En el servidor, registra las excepciones no controladas con la plataforma y la versión del shell de la sesión, para que un mismo incidente se pueda leer desde ambos extremos. Después añade un evento de uso por función del dispositivo: foto capturada, firma guardada, ubicación resuelta, notificación push recibida, subida en cola enviada, cada uno con su resultado y el shell del que procede. Dos semanas de esos datos responden a lo que las discusiones no pueden, por ejemplo si la alternativa de la cámara se activa porque se denegó un permiso o porque el shell es demasiado antiguo para tener esa capacidad.

La telemetría son datos sobre el trabajo de las personas, así que declárala. Nada de contenido de las fotos, ni imágenes de firmas, ni nombres de clientes, ni coordenadas en bruto cuando basta con un identificador de orden de trabajo; y lo que sí recojas debe figurar en la declaración de privacidad que presentaste. El proyecto final termina con una lista de comprobación de publicación, porque cuatro distribuciones tienen cuatro formas distintas de fallar y la lista es lo que hace que una publicación sea repetible por alguien que no estaba presente cuando se diseñó.

- Navegador: servidor desplegado, endpoint de versión actualizado, prueba rápida con una sesión nueva.
- PWA: versión de la caché del service worker incrementada, flujo de actualización verificado desde la compilación anterior.
- Escritorio: instalador firmado y con marca de tiempo, firma verificada en una máquina limpia, paquete de Intune sustituido.

- Móvil: claves y perfiles correctos, versión mínima del shell revisada, despliegue escalonado configurado, panel de fallos vigilado el primer día.



Regla de empaquetado

La aplicación sigue siendo una aplicación web Wisej.NET servida por tu servidor; el shell es un medio de distribución. Mantén todas las funciones operativas en un navegador normal y añade después las capacidades del dispositivo como mejoras progresivas detrás de una comprobación de capacidades.



Objetivos de aprendizaje

Al terminar este módulo serás capaz de:

- ✓ Firmar paquetes de escritorio y móviles y explicar qué demuestra cada certificado.
- ✓ Elegir entre distribución por tienda, empresarial o directa en cada plataforma y preparar cada envío.
- ✓ Diseñar una estrategia de versiones y actualizaciones en la que el servidor pueda avanzar más rápido que los shells.
- ✓ Añadir telemetría de fallos y de uso en todos los shells y entregar el proyecto final de FieldOps.

Guía del laboratorio / examen

Qué construirás en el laboratorio práctico, el enfoque sugerido y los entregables requeridos.



Laboratorio práctico

Completa el proyecto final de FieldOps: firma el instalador de Windows y los paquetes de Android e iOS, prepara una distribución empresarial para los shells móviles y un borrador de ficha para las tiendas, define una regla de compatibilidad entre las versiones del servidor y las del shell con una comprobación de versión mínima al iniciar, añade informes de fallos y un evento de uso para cada función del dispositivo, y entrega una lista de comprobación de publicación que cubra los cuatro modelos de distribución.

Punto de partida

Abres la solución FieldOps tal como la dejó el Módulo 5: el servidor Wisej.NET con la lista de órdenes de trabajo, el detalle y el flujo de estados, el manifiesto y el service worker de la PWA, el shell de escritorio WebView2, el shell .NET MAUI para Android e iOS, y el puente que ya expone la cámara, la firma, la ubicación y las notificaciones push detrás de comprobaciones de capacidades. Por ahora todo se ejecuta desde compilaciones de depuración sin firmar que apuntan a un servidor de desarrollo.

Enfoque sugerido

- ✓ Genera primero las tres identidades de firma (un certificado de firma de código para Windows, un keystore con una clave de subida registrada en Play App Signing y un certificado de distribución de Apple con un perfil de aprovisionamiento que incluya el entitlement de notificaciones push) y registra para cada una quién tiene la clave y cuándo caduca.
- ✓ Compila y firma un paquete de publicación por plataforma y después verifica cada firma como lo hará el dispositivo de destino: comprueba la firma Authenticode y su marca de tiempo en una máquina Windows limpia, y confirma que las compilaciones móviles se instalan desde su canal previsto y no desde un cable de depuración.
- ✓ Añade el protocolo de versiones antes de que se cargue nada más: expón desde la configuración la versión del servidor y la versión mínima del shell por plataforma, haz que cada shell comunique su plataforma y su versión al conectarse, y guarda ambas en la sesión.
- ✓ Construye la pantalla de actualización que muestra el filtro a un shell obsoleto, con el destino que corresponde a su canal, y un aviso descartable para un shell que simplemente va por detrás de la última compilación.
- ✓ Integra los informes de fallos en cada shell y un evento de uso en cada llamada al puente; después redacta la lista de comprobación de publicación recorriendo una publicación completa en el navegador, la PWA, el escritorio y el móvil.

Entregables requeridos

- ✓ Paquetes firmados para Windows, Android e iOS
- ✓ Configuración de la distribución empresarial y borrador de ficha para las tiendas

- ✓ Comprobación de compatibilidad de versiones entre servidor y shell
- ✓ Telemetría de fallos y de uso
- ✓ Lista de comprobación de publicación para todos los modelos de distribución

Preguntas de repaso

- ✓ El shell de Android de un técnico tiene seis semanas y el servidor se ha publicado cuatro veces desde entonces. ¿Cuáles de esos cuatro cambios del servidor deberían haber subido la versión mínima del shell, y cómo lo habrías sabido en su momento?
- ✓ El instalador de Windows se firmó pero sin marca de tiempo, y el certificado caducó el mes pasado. ¿Qué hacen las instalaciones existentes, qué hacen las descargas nuevas y cuál es el camino más corto para volver a tener un instalador de confianza?
- ✓ El evento de uso de la cámara muestra un fuerte aumento de la alternativa del navegador solo en iOS. Nombra tres explicaciones compatibles con los datos y di qué campo adicional del evento permitiría distinguirlas.



Regla de empaquetado

La aplicación sigue siendo una aplicación web Wisej.NET servida por tu servidor; el shell es un medio de distribución. Mantén todas las funciones operativas en un navegador normal y añade después las capacidades del dispositivo como mejoras progresivas detrás de una comprobación de capacidades.