OPS Auto Reminder

# Bevezetés

Az Auto Reminder példaalkalmazásban Microsoft Outlookból találkozókat importálhatunk egy kattintással, majd a találkozókban szereplő contact-okat különböző módokon értesíthetjük. A találkozó előtt megadott időben a program felhívhatja, és beolvashat egy emlékeztető szöveget, küldhet a résztvevőknek SMS-t vagy emailt az Outlookban tárolt partner névjegye alapján. A program futása alatt az időzített emlékeztetők a megfelelő időben értesítik a találkozó résztvevőit.

# Szoftverkövetelmények

Microsoft Windows 7, 8, Vista

Microsoft Visual Studio 2012, .NET 4.0

# A példaprogram használata

Induláskor megjelenik a login képernyő, a hol meg kell adni az Ozeki Phone System SDK szolgáltatásának címét és portját, az Ozeki Phone System-ben az office users menüpontjában létrehozott felhasználó nevét és jelszavát.

## Beállítások

Bejelentkezés után a settings menüpontra kattintva szükség van az alap működési beállítások megadására. A General tabon az API extension beállítás szükséges a hívásokhoz és az üzenetküldésekhez is. Az Ozeki Phone System-ben fel kell venni egy API extension-t (Extensions/Install new), amelyet ez a példaprogram fog használni, majd ennek az extensionnek az azonosítóját kell itt megadni.

Az E-mail tabon az Email settingsnél meg kell adni a küldő email címét, ez a cím fog megjelenni küldőként a kimenő üzenetekben. Megadható, hogy értesítéskor az adott személy legelső címére vagy az összes címére küldje-e el a program az üzenetet. A paraméterezhető email template-ben a névjegy adatait felhasználva a küldendő email tartalmát lehet megadni. Az SMS és a Call tabon a ugyanezek a személy telefonszámára és az SMS üzenet illetve felolvasott szöveg tartalmára beállíthatók.

## Működés

A File menü Import from Outlook… menüpontjára kattintva betöltődnek a programba az Outlookban felvett találkozók adatai. Importálás előtt az Outlookban léteznie kell egy beállított fióknak, névjegyeknek és felvett találkozóknak. Sikeres importálás után megjelennek a találkozók a főképernyőn.

Emlékeztető ezekhez a találkozókhoz a Settings menüpontban a Reminders tabon háromféle adható meg: SMS, Email és hívás. Meg kell adni, hogy a találkozó előtt mennyi idővel történjen a hívás vagy üzenetküldés. Az add majd az OK gombra kattintva az emlékeztető elmentődik. Amíg a program fut, az ütemezett hívásokat és üzenetküldéseket a program automatikusan a megfelelő időben végrehajtja, melyet a főképernyőn az emlékeztetők állapotváltozása is mutat. Amennyiben az emlékeztető időpontja a találkozót megelőző időpont, az emlékeztető egyből végrehajtódik, így a működés is azonnal tesztelhető.

# Implementáció

Az implementáció a projektben hivatkozza felhasználja a OzCommon projekteket, részben arra épülnek az AutoReminder GUI elemei. További funkciók is vannak, melyeket a közös projektekből használ a példaprogram.

## Model

A program App.xaml-jének háttérkódjában inicializálódnak induláskor a szükséges osztályok, beregisztrálódnak az IOC containerbe a megosztott komponensek, például az OutlookCalendarParser, ami kapcsolatba lép majd az Outlook-kal, a UserInfoSettingsRepository, ami a beállításokat tárolja, vagy a Client, ami az OPS SDK kliens burkoló osztálya. Innen jönnek fel hiba esetén a figyelmeztető ablakok, a LoginWindow indításkor, illetve megjelenik a MainViewModel–hez tartozó AutoReminderMainWindow bejelentkezés után.

A MainViewModel model osztályban példányosítódik a Reminder osztály, ami ütemezni fogja a beállított emlékeztetőket, kezeli a hívásokat és az üzenetküldéseket. Az InitSettings() metódusban kikéri a beállításokat, amelyek az IGenericSettingsRepository–ban tárolva vannak, ha szükséges, létrehozza az alapértékeket. Beállításokat tároló model az AppPreferences osztály, melyben le vannak tárolva a beállított emlékeztetők (ReminderActionEntry), a táblázatban megjelenő bejegyzések (ReminderActionEntry), a korábban importált találkozók, a template szövegek stb. Az InitStart() metódusban elindítja az importálási folyamatot, majd az importálás végeztével (OutlookParserOnAppointmentsParsed()) eltűnik a várakozó ablak, frissül a találkozó lista a főképernyőn és elmentődnek a találkozók a beállítások közé.

## OutlookCalendarParser

Beállításokat átadva a Start() metódusnak egy külön szálon elindul az importálás (InternalStart()). Itt az Outlook COM interfészéhez kapcsolódik, majd keresztül kikéri az aktuális találkozókat és a hozzá tartozó contact-okat. Összegyűjti a névjegyek adatait: nevek, telefonszámok, email címek. A kész adatosztályt, amely a találkozókat és a hozzá tartozó contact-ok adatait tartalmazza (Appointment osztály), a parser feldobja eseményként (AppointmentFound), illetve jelzi, hogyha a parseolás véget ért (AppointmentsParsed).

## Reminder

A Reminder osztály ütemezi a beállított emlékeztetőket, kezeli a hívásokat és az üzenetküldéseket. Konstruktorában kikéri az aktuális beállításokat, amit majd használni fog, illetve az OPS SDK burkoló osztályát. Indít egy Timert, ami bizonyos időközönként ellenőrzi a beállított remindereken végigiterálva, hogy nem érték-e el az értesítés idejét. Ha valamelyik elérte, meghívódik a InitiateReminderAction metódus, ahol az OpsClient-ből az IClient burkolatán keresztül lekéri az Ozeki Phone System-ből azt az API extension-t, aminek az azonosítóját a settings-ben megadtuk. Sikertelen lekérés esetén hibaüzenetet jelenít meg. Az értesítendő személyt felhasználva a beállítások alapján sorra veszi az értesítendő telefonszámokat és email címeket.

Hívás esetén a telefonszámok mindegyikére csinál egy hívást az API extension segítségével, majd mielőtt elindítja, feliratkozik a hívás állapotváltozás és hiba eseményeire.

var call = apiExtension.CreateCall(number.Replace(" ", ""));

…

call.Start();

Majd ezután várakozunk addig egy ManualResetEvent-tel, amíg az adott hívás be nem fejeződött.

callDone.WaitOne();

Közben elindult a hívás, és ha a hívást felveszik a másik oldalon, tehát az állapotváltozásnál InCall állapotot kap, akkor a program be fog játszani egy szöveget text to speech-csel a settings-ben megadott template alapján. Ehhez erre a hívásra rá kell csatolni egy TextToSpeech AudioHandler-t, beállítani rajta a beolvasandó szöveget, majd elindítani a beolvasást. A hang az OPS SDK media küldő rendszerén keresztül eljut az Ozeki Phone System-be, majd ott a túlsó fél hívásába bekerülnek a hangcsomagok, így hallani fogja a felolvasott szöveget.

TextToSpeech speech = new TextToSpeech();

speech.Stopped += (o, eventArgs) => call.HangUp();

call.ConnectAudioSender(speech);

var message = settings.CallTemplate;

var context = \_templateModifier.CreateCurrentContext(appointment, attendee);

message = \_templateModifier.Modify(message, context);

speech.AddAndStartText(message);

Ha a szöveget végig felolvasta a TextToSpeech, akkor a hívást bontja a program. Egy hívást befejezettnek tekintünk hiba esetén, pl. nem volt megtalálható a másik személy, ezért a hívás nem indult el. Vagy befejezett a hívás akkor is, ha olyan állapotváltozásról értesülünk, amikor a hívás véget ért, ekkor Set-eljük a ManualResetEvent –et, hogy a következő értesítendő féllel folytassa.

SMS küldés esetén sorra veszi a telefonszámokat, és ezekre a számokra kiküld egy SMS-t a beállított SMS template szöveg alapján. A template-ben lévő paraméterek behelyettesítődnek az aktuális a címzettnek az adataival. Az email értesítés is ugyanígy történik, de itt még felhasználja a beállításnál szereplő küldő címét is, amit megad az email kiküldésénél.