

Služba SMTP

Táto služba realizuje posielanie emailov cez SMTP protokol. Služba sa štartuje z adresára `D2000_EXE/bin64/smtp.exe` (alebo v prípade Linuxu: `D2000_EXE/bin64/smtp`).

Parametre príkazového riadku

Parametre služby sú nasledovné:

- `-c <connectionString>` – adresa, na ktorej poúva proces `D2Connector.exe` na prichádzajúce JAPI spojenia. Ak je JAPI spojenie šifrované, za bodkočiarkou sa nachádza cesta k súboru s certifikátom. Ak je využitá redundancia, parameter je možné zadať viackrát a vymenovať tak všetky body pripojenia. Príklady hodnoty parametra:
`server.domain.sk:3120`
`172.16.1.3:3121`
`srvapp120v:3120;certifikat.crt`
- `-w <applicationName>` – voliteľné meno služby, implicitne je to `SELF(.EMS -Email Service)`
- `--logging-directory <logging directory>` - voliteľná cesta k logovaciemu adresáru, implicitne je hodnota nastavená na relatívnu cestu `./log` do logovacieho adresára `D2000`. Pokiaľ adresár na tejto relatívnej ceste neexistuje, loguje sa do vytvoreného adresára `./log` v pracovnom adresári, pod ktorým je proces spustený
- `--reconnectTimeout <number of seconds>` - počet sekúnd, za ktorý sa proces pokúsi znova pripojiť na `D2Connector.exe`, implicitná hodnota je 10 sekúnd

Okrem parametrov zadávaných v príkazovom riadku je možné konfigurovať aj samotný JVM proces, v ktorom služba beží cez súbor `smtp.cfg`, ktorý sa musí nachádzať v tom istom adresári - `D2000_EXE/bin64/`. V súbore je možné zadať parametre [spôsobom zdokumentovaným pre Jar2Exe nástroj](#). Napríklad zapnutie ladenia celej sieovej komunikácie pre JVM je možné nasledujúcou konfiguráciou `smtp.cfg`.

```
option -Djavax.net.debug=all
```

ESL rozhranie pre posielanie mailov

Služba publikuje funkcionálnosť cez nasledovné ESL rozhranie:

I.SMTP_Service_v1

```
RPC PROCEDURE [_hTC, TC_B] OpenConnection(IN TEXT _host, IN INT _port, IN TEXT _defaultFrom, IN BOOL
_enableSsl, IN TEXT _userName, IN TEXT _password, IN BOOL _disablePlainAuthenticationSend, IN TEXT
_additionalProperties)
RPC PROCEDURE [_hTC, TC_B] OpenConnectionFromProps(IN TEXT _properties)
RPC PROCEDURE [_hTC, TC_C] SendMail(IN INT _mailId, IN TEXT _from, IN TEXT _to, IN TEXT _cc, IN TEXT _bcc, IN
TEXT _subject, IN TEXT _body)
RPC PROCEDURE [_hTC, TC_E] CloseConnection
```

Toto rozhranie založené na aplikácii definovaných konverzáciách. Konverzácia sa iniciuje volaním RPC metódy `OpenConnection` alebo alternatívne `OpenConnectionFromProps`, ktorými sa inicializuje spojenie na SMTP server. RPC metódou `SendMail` sa posiela email v rámci danej konverzácie. Ukončenie spojenia/konverzácie je realizované cez RPC metódu `CloseConnection`. Všetky volania metód sú asynchrónne. Klient musí na svojej strane implementovať nasledovné rozhranie, aby mohol byť notifikovaný o stave svojich volaní:

I.SMTP_Client_v1

```
RPC PROCEDURE [_hTC, TC_C] OnConnectionOpened
RPC PROCEDURE [_hTC, TC_E] OnConnectionClosed(IN BOOL _closedWithError, IN TEXT _errorMessage, IN TEXT
_errorDetail)
RPC PROCEDURE [_hTC, TC_C] OnMailSent(IN INT _mailId)
RPC PROCEDURE [_hTC, TC_C] OnMailSendingFailed(IN INT _mailId, IN TEXT _errorMessage, IN TEXT _errorDetail)
```

Otvorenie SMTP spojenia cez metódy `OpenConnection` a `OpenConnectionFromProps`

Metóda `OpenConnection` realizuje pripojenie na SMTP server cez parametre metódy. Interne je posielanie emailov realizované cez knižnicu `JavaMail`, ktorá sa konfiguruje cez množinu vyššie 50 [konfigurovaných parametrov](#). Väšinu z nich je potrebné explicitne zadať iba vo výnimočných prípadoch. Z tohto dôvodu RPC metóda `OpenConnection` má iba najpoužívanejšie parametre, zvyšok je možné zadať cez parameter `_additionalProperties` vo formáte [.properties súborov](#). Alternatívne je možné iniciovať SMTP spojenie iba výlučne cez tieto nastavenia metódou `OpenConnectionFromProps` cez tento parameter `_properties` s nasledovným obsahom:

Príklad mail konfigurácie

```
mail.smtp.host=mail.xyz.com
mail.smtp.port=445
mail.smtp.from= abc@xyz.com
mail.smtp.user=abc
mail.smtp.password=abcPassword
mail.smtp.ssl.enable=true

# debugovanie posielaných mailov do logu
mail.debug=true
```

Úspešné otvorenie SMTP spojenia cez metódy `OpenConnection/OpenConnectionFromProps` je notifikované volaním metódy `OnConnectionOpened` v rámci interfacu `I.SMTP_Client_v1`, ktorý musí klient implementovať. Ukončenie SMTP spojenia alebo jej zlyhania je notifikované cez metódu `OnConnectionClosed`.

Poslanie mailu - metóda `SendMail`

Metóda `SendMail` implementuje posielanie emailu. Prvý parameter `_mailId` označuje identifikátor mailu ktorý vyplní klient pri volaní `SendMail` metódy. O úspechu resp. zlyhaní poslania mailu je klient notifikovaný cez metódy `OnMailSent` a `OnMailSendingFailed` rozhrania `I.SMTP_Client_v1`.

Príklad použitia ESL rozhrania na posielanie mailov

Nasledovný kód predstavuje príklad jednoduchej mailovej služby v aplikácii. Pre poslanie mailu staí zavola RPC procedúru `SendMail`. Event `E.MailService` interne používa konverzáciu s procesom `SELF.EMS`, ktorú udržiava otvorenú a pri rozpade spojenia ju automaticky obnoví. Zároveň, aby nedošlo k zahodeniu nejakého mailu, v kontajneri uchováva všetky doteraz neodoslané maily a pri opätovnom nadviazaní spojenia s procesom `SELF.EMS` sa ich automaticky pokúsi znova odoslať.

E.MailService

```
IMPLEMENTATION I.SMTP_Client_v1

INT _hCnt, _hTC, _nextMailId
BOOL _hasTC

; Po nadviazaní spojenia so SMTP serverom sa pokúsi odoslať všetky doteraz neodoslané maily
IMPLEMENTATION RPC PROCEDURE [_hTC, TC_C] I.SMTP_Client_v1^OnConnectionOpened
    RECORD NOALIAS (SD.MailData) _mail
    INT _count, _i

    _hasTC := @TRUE
    CNT_CNVTOARRAY _hCnt
    CNT_GETNR _hCnt, _count
    FOR _i = 1 TO _count DO_LOOP
        CNT_GETITEM _hCnt, _i, _mail
        CALL Internal_SendMail(_mail)
    END_LOOP
END OnConnectionOpened

IMPLEMENTATION RPC PROCEDURE [_hTC, TC_E] I.SMTP_Client_v1^OnConnectionClosed(IN BOOL _closedWithError, IN TEXT _errorMessage, IN TEXT _errorDetail)
    IF _closedWithError THEN
        LOGEX _errorMessage
    ENDIF
    CALL [_hTC] I.SMTP_Service_v1^CloseConnection
    _hasTC := @FALSE
END OnConnectionClosed

; Pri úspešnom odoslaní mailu ho vymaže z kontajneru
IMPLEMENTATION RPC PROCEDURE [_hTC, TC_C] I.SMTP_Client_v1^OnMailSent(IN INT _mailId)
    CNT_DELETE _hCnt, _mailId
END OnMailSent

; Pri neúspešnom odoslaní mailu zaloguje chybu
IMPLEMENTATION RPC PROCEDURE [_hTC, TC_C] I.SMTP_Client_v1^OnMailSendingFailed(IN INT _mailId, IN TEXT
```

```

_errorMessage, IN TEXT _errorDetail)
    LOGEX _errorMessage
END OnMailSendingFailed

RPC PROCEDURE [_hTC, ERROR] OnConversationAborted
    _hasTC := @FALSE
END OnConversationAborted

; Nadviazanie spojenia so SMTP serverom, ak už neexistuje
PROCEDURE Internal_InitConnection
    IF !_hasTC THEN
        CALL [(0)] I.SMTP_Service_v1^OpenConnection("mail.example.com", 25, "", @FALSE, "user", "secret", @FALSE,
        "") ASYNC ON SELF.EMS TC_B _hTC
    ENDIF
END Internal_InitConnection

PROCEDURE Internal_SendMail(IN RECORD NOALIAS (SD.MailData) _mail)
    TEXT _from
    _from := "no-reply@ipesoft.com"
    CALL [_hTC] I.SMTP_Service_v1^SendMail(_mail[1]^id, _from, _mail[1]^to, "", "", _mail[1]^subject, _mail[1]
^message) ASYNC TC_C
END Internal_SendMail

; RPC volaná "z vonku"
RPC PROCEDURE SendMail(IN TEXT _to, IN TEXT _subject, IN TEXT _message)
    RECORD NOALIAS (SD.MailData) _mail

    _mail[1]^id := _nextMailId
    _mail[1]^to := _to
    _mail[1]^subject := _subject
    _mail[1]^message := _message
    CNT_INSERT _hCnt, _mail[1]^id, _mail
    _nextMailId := _nextMailId + 1

    CALL Internal_InitConnection
    IF _hasTC THEN
        CALL Internal_SendMail(_mail)
    ENDIF
END SendMail

; Inicializácia as - vytvorenie kontajnera a konverzácie
BEGIN
    CNT_CREATE _hCnt
    _hasTC := @FALSE
    _nextMailId := 1
    CALL Internal_InitConnection
END

```