

Objectif

Développer un serveur intermédiaire capable de communiquer et de gérer le Smart Coin System. Ce serveur doit recevoir les événements de la machine et exécuter les actions demandées.

Développeur

⑩ **Nom** : Ahmed Douar

⑩ **Date de développement V3.2.0** : 17/10/2025

Résumé

Développement d'un serveur en Node.js utilisant le SDK d'ITL. Ce serveur agit comme intermédiaire, en ajoutant la possibilité de contrôler la machine via MQTT.

Fonctionnalités

- ⑩ Système asynchrone
- ⑩ Système qui journalise toutes les transactions
- ⑩ Système documenté (référence : documentation ITL)
- ⑩ Système avec une gestion complète d'authentification
- ⑩ Système qui écoute les événements simultanément NV22 (cash) et NV150 (coin)
- ⑩ Architecture basée sur MQTT pour la diffusion des événements d'état et des transactions.

Extra

- J'ai implémenté des fonctions **quality of life** qui facilitent la gestion des appareils :
 - **quickConnect** : ouvre la connexion avec tous les appareils.
 - **quickDisconnect** : ferme la connexion avec tous les appareils.
 - **enable_cv** : démarre tous les appareils.
 - **disable_cv** : arrête tous les appareils.
 - **payout** : rendre la monnaie automatiquement (prioriser les billets).

Exemples

```
*{  
  "action": " quickConnect " | " quickDisconnect "  
}
```

```
*{  
  "action": " enable_cv " | " disable_cv "  
}
```

```
*{  
  "action": " payout ",  
  "data": "5500"(55.00 euro)  
}
```

Améliorations possibles

⑩ Optimisation du code et des variables pour de meilleures performances

Protocole de communication

SSP

Structure des commandes MQTT

```
{  
  "action": "",  
  "data": {},  
  "device": "cash" // "cash" ← NV22, "coin" ← NV150  
}
```

Exemples

```
{  
  "action": "getalllevels",  
  "data": {},  
  "device": "cash"  
}
```

---> Cette requête retourne tous les billets/jetons en inventaire

Requêtes à effectuer après le démarrage du serveur

```
{  
  "action": " quickConnect "  
}
```

→ cette requête ouvre la connexion avec tous les appareils.

Pour le dispositif de billets (cash)

```
{  
  "action": "openconnection",  
  "data": {  
    "ComPort": "/dev/cash",  
    "SspAddress": 0,  
    "LogFilePath": "/chemin/vers/SmartCash_Log.log",  
    "SetInhibits": [],  
    "SetRoutes": [  
      { "Denomination": "500 EUR", "Route": 1 },  
      { "Denomination": "1000 EUR", "Route": 1 },  
      { "Denomination": "2000 EUR", "Route": 1 },  
      { "Denomination": "5000 EUR", "Route": 1 },  
      { "Denomination": "10000 EUR", "Route": 0 },  
      { "Denomination": "20000 EUR", "Route": 0 },  
      { "Denomination": "50000 EUR", "Route": 0 }  
    ],  
    "EnableAcceptor": true,  
    "EnableAutoAcceptEscrow": true,  
    "EnablePayout": true  
  },  
  "device": "cash"}
```

Pour le dispositif de pièces (coin)

```
{  
  "action": "openconnection",  
  "data": {  
    "ComPort": "/dev/coins",  
    "SspAddress": 16,  
    "LogFilePath": "/chemin/vers/SmartCoin_Log.log",  
    "SetInhibits": [],  
    "SetFeederRoutes": [  
      { "Denomination": "5 EUR", "Route": 7 },  
      { "Denomination": "10 EUR", "Route": 7 },  
      { "Denomination": "20 EUR", "Route": 7 },  
      { "Denomination": "50 EUR", "Route": 7 },  
      { "Denomination": "100 EUR", "Route": 7 },  
      { "Denomination": "200 EUR", "Route": 7 }  
    ],  
    "SetCashBoxPayoutLimit": [100, 50, 40, 30, 20, 10, 5, 0],  
    "EnableAcceptor": true,  
    "EnablePayout": true  
  },  
  "device": "coin"}
```

ROUTE CODE:

0: cashbox.

1:recycler.

7:payout box.

Notes d'implémentation

- ⑩ Les chemins des ports série (ComPort) et des fichiers de log (LogFilePath) doivent être adaptés à l'environnement de déploiement
- ⑩ Le système utilise MQTT pour la communication asynchrone
- ⑩ Toutes les transactions sont journalisées pour le suivi et le débogage
- ⑩ L'authentification est gérée pour sécuriser les communications
- ⑩ Pour plus d'information sur les commands veuillez consulter doc itl sdk.