BIO-METRIC DEVICE INTEGRATION

- **STARTEK (FM220U)**
- **MANTRA (MFS100)**
- **MORPHO (MSO1300 E3)**
-

Device Information

```
  getDeviceInformation(device: 'mantra' | 'morpho' | 'startek') {
    let deviceInfo = device == 'mantra' ? "/rd/info" : device == 'startek' ? '/rd/info' : '/getDeviceInfo';
    let URL = device == 'mantra' ? mantraURL : device == 'startek' ? startekURL : morphoURL
    return this.http.request('DEVICEINFO', URL + deviceInfo, { responseType: "text" }).pipe(
      switchMap(async xml => await this.parseXmlToJson(xml))
    );
  }
```

Sample Response

```
{
  "RDService": {
      "$": {
          "xmlns:xsi": "http://www.w3.org/2001/XMLSchema-instance",
          "xmlns:xsd": "http://www.w3.org/2001/XMLSchema",
          "status": "READY",
          "info": "RD service for Startek FM220 provided by Access Computech"
      },
      "Interface": [
          {
              "$": {
                  "id": "CAPTURE",
                  "path": "/rd/capture"
              }
          },
          {
              "$": {
                  "id": "DEVICEINFO",
                  "path": "/rd/info"
```

Device Status

```
      getDeviceStatus(device: 'mantra' | 'morpho' | 'startek'):
Observable<any> {
    let URL = device == 'mantra' ? mantraURL : device == 'startek' ?
startekURL : morphoURL
    return this.http.request('RDSERVICE', URL, { responseType:
"text" }).pipe(
      switchMap(async xml => await this.parseXmlToJson(xml))
    );
  }
```

Sample Response :

```
{
  "$": {
      "xmlns:xsi": "http://www.w3.org/2001/XMLSchema-instance",
      "xmlns:xsd": "http://www.w3.org/2001/XMLSchema",
      "dpId": "STARTEK.ACPL",
      "rdsId": "ACPL.WIN.001",
      "rdsVer": "1.0.4",
      "dc": "",
      "mi": "FM220U",
      "mc": "",
      "error": "Pl connect FM220"
  }
}
```

Capture Fingerprint

```
 captureFinger(device: 'mantra' | 'morpho' | 'startek', request:
any): Observable<any> {
    let URL = device == 'mantra' ? mantraURL : device == 'startek' ?
startekURL : morphoURL
```

```
    let params = `<?xml version="1.0"?> <PidOptions
ver="${request?.pidOptVer}"> <Opts fCount="${request?.fCount}"
fType="${request?.fType}" iCount="${request?.iCount}"
pCount="${request?.pCount}" format="${request?.format}"
pidVer="${request?.pidVer}" timeout="${request?.timeout}"
posh="${request?.posh}" env="${request?.env}" /> <CustOpts> <Param
name="mantrakey" value="undefined" /> </CustOpts> </PidOptions>`
    return this.http.request('CAPTURE', URL + "/rd/capture", { body:
params, responseType: "text" }).pipe(
      switchMap(async xml => await this.parseXmlToJson(xml))
    );
  }
```

XML Parser

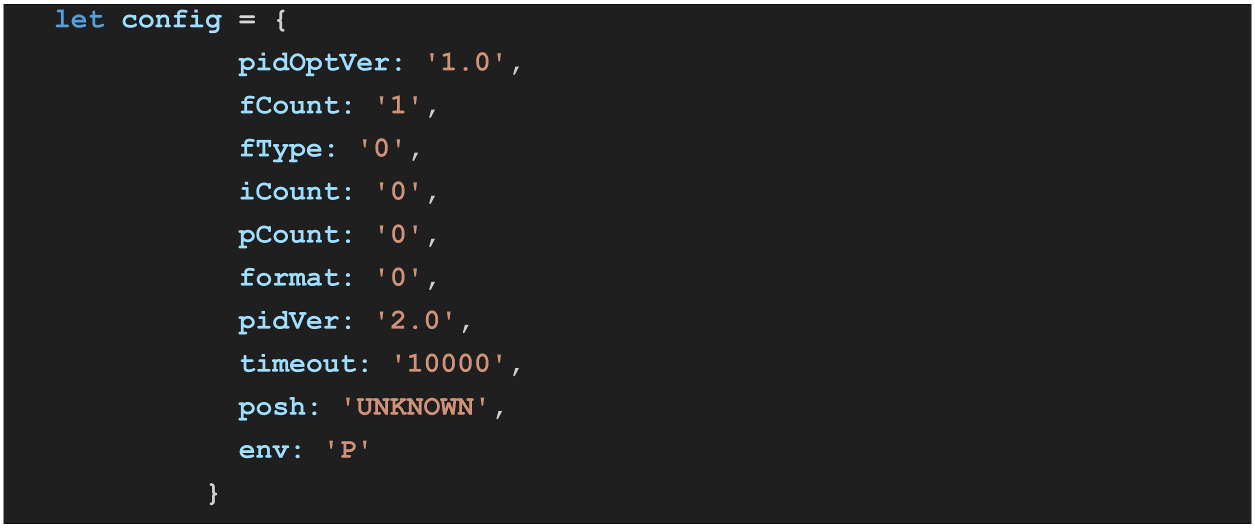
```
  async parseXmlToJson(xml: any) {
    return await xml2js
      .parseStringPromise(xml, { explicitArray: false })
      .then(response => response);
  }
```

Dependencies

1. **xml2js: ^0.6.2**
2. **@types/xml2js :** ^0.4.12

Localhost URL :

```
const mantraURL = "http://localhost:11100"
const morphoURL = 'http://127.0.0.1:11101'
const startekURL = 'http://localhost:11100'
```



```
let config = {
           pidOptVer: '1.0',
           fCount: '1',
           fType: '0',
           iCount: '0',
           pCount: '0',
           format: '0',
           pidVer: '2.0',
           timeout: '10000',
           posh: 'UNKNOWN',
           env: 'P'
         }
```