Varsanyi's Unique Communication Interface Rule

Magyar nyelvű technikai dokumentáció a **VUCIR** működéséről.

Weboldal: https://v01it.hu/ GitHub: https://github.com/varsanyib

Tartalomjegyzék

1 Bevezetés

A **VUCIR** rövidítéssel ellátott egyéni kommunikációs szabvány a készítő által létrehozott szerveroldali és felhasználó oldali alkalmazások közötti közös illesztőfelület struktúráját és kezelését tartalmazza. A „protokoll” használatával egyszerűbbé teheti a projekt és csapatmunkák közötti egyszerű együttműködést, amely nem csak időtakarékossá teszi a fejlesztést és a kommunikációt, hanem csökkenti a hibalehetőségeket is. A jelen technikai dokumentáció nem tér ki a „RESTful Application Programming Interface”, röviden „API” alapjainak működésére, kizárólag az egyéni szabvány részletei kerültek kidolgozásra.

2 Alapok

2.1 Formátum

A szabvány kizárólag „JavaScript Object Notation”, röviden „JSON” formátumban jeleníti meg a szerveroldali alkalmazás által feldolgozott információkat, amelyeket az RFC 4627 és RFC 8259 szabványok definiálnak.

A „backend” fejlesztésnél használt programozási nyelveknél fontos, hogy a felületen megjelenített adat „MIME” típusa „application/json” formátumra kerüljön beállításra, mivel az „API” tesztelésére használt alkalmazások, (például: „Postman Client”) a beállítást felismerve, automatikusan olvasható állapotban jeleníti meg az adatokat, amelyek segítik a megjelenítést.

A legtöbb tesztelésre használható alkalmazás automatikusan formázza a megjelenített adatokat, ezért nem kerülnek felhasználásra - a programozási nyelvektől függően - olyan kiegészítők, amelyek a már megjelenített adatot kényszerítik formázásra. (Például: „JSON Pretty Print”)

2.2 Struktúra és státuszkód

A **VUCIR** egyik fő működési elve a dokumentált szabványok struktúrájának használata. Ennek segítségével, ha a projektben megosztott a „backend” és „frontend” fejlesztők munkája, akkor az adatok kezelése leegyszerűsíthető.

Egy lekérdezés során, a fő információk alapján az érdemleges további adatok feldolgozása egyszerűbben kezelhető, ha tudjuk már előre a lekérdezés eredményét, és ezt nem szükséges külön implementálni egy asztali, mobil vagy akár egy webes alkalmazásban.

A lekérdezés sikerességét nem minden esetben a „HTTP” státuszkódok határozhatják meg, de egy végrehajtásnál elengedhetetlen ennek a vizsgálata. A státuszkódokhoz nem szükséges további megerősítő struktúrát hozzárendelni, hiszen már a lépcsőzetes tesztelésnél egy lekérdezés sikeressége itt is megbukhat.

Státuszkód	Hibaüzenet	Lehetséges hiba	Példa
404	Not Found	A megadott hivatkozáson elképzelhető, hogy nem található lekérdezés, vagy elírásra került a link.	https://pelda.intra/szevreridо
500	Internal Server Error	Belső szerverhiba, amely lehet egy proxy vagy adatbáziskapcsolati hiba	https://pelda.intra/szerverido

2.3 Meghatározó sikerességi eredmény

A „HTTP” státuszkódok alapján, a 200-as kód a megfelelő sikeres lekérdezést jelentheti, viszont ez nem jelentheti teljes mértékben, hogy logikailag is teljesült a lekérdezés, amiből a végfelhasználó a termékben (szoftverben) már megfelelő adatokhoz juthat hozzá.

A **VUCIR** működésében az összes olyan lekérdezés, amely nem tartalmaz belső hibát, illetve létező hivatkozáson lévő lekérdezést próbál a felhasználó megvalósítani, az a megfelelő 200-as kódot megkaphatja.

A meghatározó eredmény sikerességét a „backend” fejlesztő határozza meg. Ez minden esetben egy logikai értékként használandó, amely a felhasználónak röviden, tömören „megfogalmazza”, hogy az adott lekérdezés a számára sikeresnek mondható, vagy nem. (Például: A felhasználónak nincs jogosultsága megtekinteni egy adatcsomagot és ez esetben a meghatározó sikerességi eredmény értéke hamis lesz.)

Ez a logikai változó eldöntheti, hogy a felhasználó részére egy hibaüzenet kerüljön megjelenítésre egy alkalmazásban, vagy a sikeres lekérdezést tartalmazó eredmény.

Meghatározó sikerességi eredmény	Lehetséges üzenet
hamis	Nincs megfelelő jogosultsága az adatok eléréséhez!
igaz	Az Ön születési ideje 2000. szeptember 1.

2.4 Üzenetek

A rendszerben többféle kidolgozási módszert jeleníthetünk meg, amely a meghatározó sikerességi eredménytől függenek. Amennyiben az eredmény hamis, akkor a megjelenítési mód már csak attól függ, hogy a **VUCIR** melyik változatának használatát kezdeményezi a fejlesztő vagy a fejlesztőcsapat melyik verzióban állapodtak meg a tervezési folyamat elején.

A különböző változatok között az alkalmazás nagysága és támogatottsága alapján érdemes választani.

3 Verziók

A **VUCIR** többféle módszert ajánl a fejlesztési projektek létrejöttéhez. A projekt nagysága meghatározhatja, hogy melyik módszer használható, illetve milyen érdemi területhez köthető az „API” válaszok megfeleltetése. (Például: Eltérő fejlesztési mód szükséges egy vállalati rendszerhez, vagy egy otthoni okosotthon vezérlő modul fejlesztéséhez, szerver és kliensoldalon egyaránt.)

A módszerek megkülönböztetése az üzenetek kezelésén tér el, az adatok feldolgozása, illetve megjelenítése nem tér el, az alapokban foglaltaktól.

Az üzenetek természetesen kiegészíthetőek olyan plusz információkkal, amelyek a további fejlesztési munkálatokat segítik. (Például: Üzenet egyéni színe, speciális hangjelzés kódja.)

3.1 VUCIR-FIX módszer

A felhasználó az üzenetet a fejlesztőtől közvetlenül kapja meg, amely lehetővé teszi, hogy minden különböző lekérdezésnél különböző tartalmak is megadhatóak legyenek. Egyes szemantikai hibakezeléshez egyéni üzenet hozzáadható, amely megadhatja a pontos hibát, annak javításának lehetőségét.

Felhasználó által okozott hiba	Lehetséges üzenet
Születési idő megadásánál a felhasználó 12.-nél nagyobb értékű hónapot kíván megadni.	A születési időben a hónapnak 1-12 intervallum közé kell esnie! (január-december)
Születési idő megadásánál a felhasználó negatív számot adott meg az év értéknek.	A születési időben az év nem lehet negatív érték!

3.2 VUCIR-MULTIFIX módszer

A „FIX” módszerhez hasonlóképpen a fejlesztő határozza meg, az összes lehetséges eredmény kimenetét, viszont az üzenetek több nyelven kerülnek megjelenítésre. Azokra a nyelvekre kell lefordítani az üzeneteket, amelyek az „API” támogatásában szerepel.

Felhasználó által okozott hiba	Nyelv	Lehetséges üzenet
A megadott felhasználónév vagy jelszó nem található a rendszerben, vagy hibás.	magyar	Hibás felhasználónév vagy jelszó!
	angol	Incorrect username or password!
	német	Benutzername oder Passwort falsch!
	orosz	Неверное имя пользователя или пароль!
	szlovák	Nesprávne užívateľské meno alebo heslo!
	szerb	Погрешно корисничко име или лозинка!

3.3 VUCIR-EXT módszer

Az „EXT” módszer esetén az üzenetek szövege egy külső tárhelyről kerül importálásra, amely lehet egy fájl vagy egy plusz lekérdezés, esetleg szolgáltatás. Az összes üzenethez a fejlesztő kódszavakat, azonosítókat rendel, amely során a lekérdezésekhez elegendő a kódszavakat hozzárendelni.

Fontos, hogy az „API” jelenítse meg a kódszavakat is a lekérdezésben, amely segítségével a „frontend” fejlesztők további kezeléseket rendelhetnek egy kódszóhoz. (Például: „LOGIN_ERR1” kódszóval ellátott üzenet, a „Hibás felhasználónév vagy jelszó!” mondatra mutat, és így a „frontend” fejlesztők hozzárendelhetnek eseményeket, animációkat, színeket.)

Kódszó (azonosító)	Felhasználó által okozott hiba	Lehetséges üzenet
LOGIN_ERR1	Elrontott felhasználónév vagy jelszó.	Hibás felhasználónév vagy jelszó!
	Üres a jelszó mező értéke.	Hibás felhasználónév vagy jelszó!
LOGIN_ERR2	A felhasználói adatok ellenőrzése sikeres, viszont a fiók manuálisan letiltásra került.	Az Ön fiókja jelenleg le van tiltva! Forduljon az IT csoportjához!
	A felhasználói adatok ellenőrzése sikeres, viszont a fiók automatikusan letiltásre került, más országból való belépés miatt.	Az Ön fiókja jelenleg le van tiltva! Forduljon az IT csoportjához!
LOGIN_INFO	-	Sikeres bejelentkezés!
LOGIN_NOTICE	A belépést követően a felhasználónak módosítania kell a jelszavát.	Kérjük változtassa meg a jelszavát!

3.4 VUCIR-MULTIEXT módszer

Az „EXT” módszer kiegészítéseként, az üzenetek több nyelven kerülnek megjelenítésre a lekérdezésben. A „MULTIFIX” módszerhez hasonlóan az „API” támogatásában feltüntetett nyelvekre kell lefordítani az üzeneteket a külső adatforrásban, a kódszavak változása nélkül. Ez a módszer jelentősen könnyíti a plusz nyelvek hozzáadását a támogatásban, mivel csak a külső adatforrásban szükséges az üzenetek lefordítása, amely nem igényel különösebb programozói, szoftverfejlesztéshez kapcsolódó és összefüggő tudást.

Kódszó (azonosító)	Nyelv	Lehetséges üzenet
LOGIN_ERR1	magyar	Hibás felhasználónév vagy jelszó!
	angol	Incorrect username or password!
	orosz	Неверное имя пользователя или пароль!
	szlovák	Nesprávne užívateľské meno alebo heslo!

4 Összeállítás

Minden „API” lekérdezésre adott válasznál az összeállítás elve a legfontosabb, ettől eltérni nem lehetséges. Az alap struktúrák leegyszerűsítik a további fejlesztési folyamatokat, gyorsabbá teszik a lekérdezések feldolgozását.

4.1 Alap struktúra

Az alap struktúrába tartozik a meghatározó sikerességi eredmény logikai változó értéke, a lekérdezés eredménye, a megjelenített üzenet, illetve a lekérdezés időpontja időbélyeg („timestamp”) formátumban.

Példa	Meghatározó sikerességi eredmény	Eredmény	Üzenet	Időbélyeg
Sikeres bejelentkezés.	igaz	Név: Fiktív Péter Életkor: 20	Sikeres bejelentkezés!	1074985200
Elrontott jelszó bejelentkezésnél.	hamis	„null”	Hibás felhasználónév vagy jelszó!	1703597695

Ha egy lekérdezés sikeres, és ezt megerősíti a meghatározó sikerességi eredmény is, akkor az adatszolgáltatás feldolgozható, így az eredmények megjeleníthetőek a felhasználó számára a kész termékben. (opcionális)

Ezt az implementálás alatt a fejlesztőnek egy egyszerű feltételvizsgálat alapján meg tudja határozni, hogy a lekérdezés eredményénél a hibaüzenet adatait kell feldolgoznia (és megjelenítenie) vagy a további információkat.

Célszerű a webszervert, illetve az egyéb kommunikációs platformok alapvető hibaüzeneteit is már ebben az adatszerkezetben megjeleníteni, így még több olyan hibalehetőség kerülhető el egy kész szoftverben, amely akadályozná a működést.

A lekérdezésekben lehetőség van megjeleníteni az adatbázis vagy a szerveroldali alkalmazás pontos időpontját, így felhasználható és kiszámolható egy alkalmazásban az eltérő időzóna, vagy adatfrissítések időpontja, amely további segítséget nyújthat a fejlesztés idején.

Az úgynevezett „Unix Timestamp”, vagyis időbélyeg megadja pontosan, hogy a referencia időpont óra mennyi másodperc telt el. A referencia időpont 1970. január 1. 00:00:00. Az időbélyeg érték könnyen átalakítható bármilyen régióban használt időformátumra.

4.2 Támogatás megjelenítése

Amennyiben a fejlesztő vagy csapata a **VUCIR** használata mellett döntenek, akkor az „API” fő (ún. gyökér) hivatkozásán, alkalmazni kell egy teszt struktúrát. Eredményként feltüntethető, az „API” verziószáma, utolsó frissítés dátuma, készítők adatai, szabvány bevezetésének időpontja. (Például: https://api.example.com)

Ezek technikai szempontból is alkalmazhatóak, hiszen egy kliensoldali alkalmazás használatakor, érdemes ellenőrizni, hogy a „beégetett API hivatkozásán” megfelelő tartalom működik, és az alkalmazás felkészült a **VUCIR** által felépített adatok és üzenet kezelésében.

4.3 Önteszt (gyors diagnosztika)

Az „API” egyik elengedhetetlen része, az önteszt, amely során egy kliensoldali alkalmazás ellenőrizheti a szerveroldali komponensek működését. A teszt során az összes egyéni komponens ellenőrzésére lehetőséget kell adni, különböző műveletek segítségével.

Példa technológiák	Lehetséges teszt funkció
MySQL PostgreSQL	Aritmetikai művelet megoldása vagy adatbázis aktuális időpontjának lekérdezése
Redis	Teszt adat eltárolása, majd törlése
InfluxDB	Utolsó teszt időpontjának lekérdezése, majd eltárolása

5 Példák

A struktúrában használt kötelező kulcsszavak definiálják a szabvány alapjait. Ezektől eltérni nem lehetséges.

A „success” a meghatározó sikerességi eredményét, a „result” a rendszer által feldolgozott értékeket, a „message” a lekérdezésekhez tartozó üzenetet, a „timestamp” pedig a lekérdezés aktuális idejét jeleníti meg.

5.1 VUCIR-FIX módszer példája

Sikertelen bejelentkezésnél az eredmény értéke „null” és a „beégetett” szöveg lesz az üzenet értéke, amely egy „frontend” fejlesztésnél egyszerűen kiírható a szoftverben, a meghatározó sikerességi eredmény változó megvizsgálásával.

```
{
  "success":false,
  "result":null,
  "message":"Hibás felhasználónév vagy jelszó!",
  "timestamp":1703545200
}
```

Sikeres lekérdezésnél az üzenet úgyszintén megjeleníthető a szoftverben, de az adatbázis lekérdezés vagy egyéb szolgáltatásokból lekérdezett és összeállított adat megtalálható az eredmény mezőjének az értékeként.

```
{
  "success":true,
  "result":{
    "name":"Fiktív Péter",
    "age":20,
    "token":"ABCDEFGHIJKLMNOPRSTUVWXYZ987654321",
  },
  "message":"Sikeres bejelentkezés!",
  "timestamp":1703545200
}
```

5.2 VUCIR-MULTIFIX módszer példája

A specifikációban és az „API” által támogatott nyelveken elérhetőek az üzenetek további fordításai, amely a „text” értékeiben találhatóak meg. Opcionálisan hozzáadható az üzenet („message”) objektumhoz olyan értékek, amelyek fontos tényezők lehetnek a „frontend” fejlesztőknek. (Például: Hiba esetén a „color” értéke legyen a hibaüzenetet tartalmazó megjelenő ablak háttérszínének színkódja.)

```
{
  "success":false,
  "result":null,
  "message":{
    "text":[
      {
        "hu":"Hibás felhasználónév vagy jelszó!"
      },
      {
        "en":"Incorrect username or password!"
      },
      {
        "rus":"Неверное имя пользователя или пароль!"
      },
      {
        "sk":"Nesprávne užívateľské meno alebo heslo!"
      }
    ],
    "color":"#FF0000"
  },
  "timestamp":1703545200
}
```

5.3 VUCIR-EXT módszer példája

Az „API” válaszában nem sok eltérés tapasztalható, az üzenetet ellátó kódszó vagy azonosító kerül megjelenítésre az üzenetekben. A szerveroldali alkalmazásban lévő kódszó vagy azonosító alapján egy külső adatforrásból kerül az üzenet kiolvasásra és megjelenítésre a válaszban.

```
{
  "success":false,
  "result":null,
  "message":{
    "id":"LOGIN_ERR1",
    "text":"Hibás felhasználónév vagy jelszó!"
  },
  "timestamp":1703545200
}
```

5.4 VUCIR-MULTIEXT módszer példája

A „MULTIFIX” és „EXT” módszer egyesítése úgyszintén az „API” válaszában csak a megjelenő kódszóval vagy azonosítóval kerül kiegészítésre a lekérdezés.

A szerveroldali alkalmazásban az azonosítóhoz tartozó összes fordítást meg kell jeleníteni a lekérdezésben, amely szerepel az „API” támogatásában vagy a specifikációban.


```
{
  "success":false,
  "result":null,
  "message":{
    "id":"LOGIN_ERR1",
    "text":[
      {
        "hu":"Hibás felhasználónév vagy jelszó!"
      },
      {
        "en":"Incorrect username or password!"
      },
      {
        "rus":"Неверное имя пользователя или пароль!"
      },
      {
        "sk":"Nesprávne užívateľské meno alebo heslo!"
      }
    ]
  },
  "timestamp":1703545200
}
```

5.5 Támogatással kapcsolatos információk példája

A támogatással kapcsolatos információk között megtalálható, az alkalmazott **VUCIR** szabvány választott módszere, „API” verziószáma és támogatás lejáratának időpontja, fejlesztők nevei. A támogatott nyelvek listája mindenképpen megjelenítésre kerül („FIX” és „EXT” módszerekben is).


```
{
  "success":true,
  "result":{
    "standard":"VUCIR",
    "mode":"MULTIEXT",
    "api_version":"1.0.0",
    "end_of_support":1893452399,
    "supported_languages":[
      "hu",
      "en"
    ],
    "created_by":[
      "MF IT",
      "Varsányi Barnabás"
    ]
  },
  "message":{
    "id":"INFO",
    "text":[
      {
        "hu":"VUCIR szabvány alapján készült."
      },
      {
        "en":"Made based on the VUCIR standard."
      }
    ]
  },
  "timestamp":1703545200
}
```

5.6 Önteszt példája

Az önteszt során az összes létező komponens ellenőrzésre kerül, majd az eredménye logikai formában kerül megjelenítésre a lekérdezésben. Amennyiben az összes komponens megfelelően fut, akkor a meghatározó sikerességi eredmény értéke is igaz lesz.


```
{
  "success":true,
  "result":{
    "nodejs":true,
    "mysql":true,
    "redis":true
  },
  "message":{
    "id":"SELFTEST_OK",
    "text":[
      {
        "hu":"Az önteszt sikeres!"
      },
      {
        "en":"The self-test is successful!"
      }
    ]
  },
  "timestamp":1703605283
}
```


6 Összefoglalás

A szabvány elsősorban a készítő (Varsányi Barnabás) által használt megszokott lépéseket tartalmazza, amelyeket a saját projektekben való felhasználásra, egységesítésére lett elkészítve. A dokumentum az elmúlt évek tapasztalataiból került kidolgozásra, további felhasználása, szerkesztése engedélyköteles. A más projektekben való logika felhasználása nem tilos, viszont felelősséget az adatok megfelelő megjelenéséért és kezeléséért az adott projektnek a vezetője és fejlesztői/fejlesztője vállalja.

7 Felhasznált tartalmak és segédanyagok

Felhasznált tartalom	Forrás
RFC 8259	https://datatracker.ietf.org/doc/html/rfc8259
RFC 4627	https://datatracker.ietf.org/doc/html/rfc4627
JSON formázás és ellenőrzés	https://jsonformatter.curiousconcept.com/
Fordítás	https://translate.google.com/