

## Obsah

1. Metody.....	2
struct getHash(string login).....	2
struct login(string session_id, string password, string software_key).....	2
struct logout(string session_id).....	2
struct version().....	3
struct addEditCar(string session_id, struct car_data ).....	3
struct getCar(string session_id, int car_id).....	7
struct getCarId(string session_id, string custom_id).....	7
struct delCar(string session_id, int car_id).....	8
struct listOfCars(string session_id).....	8
struct listOfAllCars(string session_id, int client_id).....	8
struct addEditPhoto(string session_id, int car_id, struct photo_data).....	9
struct delPhoto(string session_id, int photo_id).....	10
struct getPhotoId(string session_id, int car_id, string client_photo_id).....	10
struct listOfPhotos(string session_id).....	11
struct listOfPhotos(string session_id, int car_id).....	11
struct addEquipment(string session_id, int car_id, array int equipment).....	11
struct listOfEquipment(string session_id, int car_id).....	12
2. Popis rozhraní.....	12
Kódování.....	12
Přihlášení.....	12
Typický průběh komunikace.....	13
3. Chybové hlášky.....	13
4. Příklady.....	14



## struct version()

- **Vrátí verzi xmlrpc serveru.**

**Návratová hodnota:**

```
struct{
    int status                status (200=OK)
    string status_message    slovní popis statusu
    struct output{
        string version       řetězec s číslem verze import. rozhraní
    }
}
```

## struct addEditCar(string session\_id, struct car\_data )

- **Vložení nebo editace inzerátu (vozidla).**

Pokud bude car\_id v car\_data kladné (nenulové) celé číslo, tak se provede editace inzerátu se shodným car\_id. Pokud bude car\_id nula, tak se provede vložení nového inzerátu. Pokud bude u updatu nebo insertu nového auta uveden „region\_nuts“ jako prázdný řetězec ve struktuře car\_data, nastaví se automaticky region\_nuts od klienta. Parametry made\_date a run\_date nesmí udávat datum starší než 2 roky a parametr tachometr (počet najetých km/mil) nesmí být větší než 5000 v případě, že parametr condition = 1 (nové).

**Parametry:**

```
string session_id          id session získané z metody getHash()
struct car_data{
    int car_id              data vkládaného inzerátu (jednotlivé položky a jejich hodnoty)
    string custom_id       idéčko inzerátu (vozidla) podle číslování sauta
    string kind_id         [řetězec] idéčko inzerátu (vozidla) podle číslování klienta
    string manufacturer_id [celé číslo] id druhu vozidla
    string model_id        [celé číslo] id výrobce
    string body_id         [celé číslo] id modelu
    string car_status      [celé číslo] id karoserie
    string color           [celé číslo] aktivni/neaktivni
    string vin             [řetězec] barva
    string state_id        [řetězec][17] vin kód 17 znaků
    string fuel            [celé číslo] id státu, ve kterém bylo vozidlo koupeno
    string tachometr       [celé číslo] druh pohoných hmot
    string tachometr_unit  [celé číslo] stav tachometru (počet najetých km, mil)
    string made_date       [celé číslo] jednotka, ve které se udává stav tachometru
    string condition       [rm datum] rok [měsíc, den] výroby vozidla
    string price           [celé číslo] stav vozidla
    string price_leasing  [celé číslo] cena vozidla; pokud je vozidlo na leasing je tu uloženo
    string engine_volume  [celé číslo] odstupné
    string dph             [celé číslo] cena na leasing
    string vat_deductable [celé číslo] obsah motoru
    string stk_date       [celé číslo] určuje zda je cena vozidla s DPH nebo bez DPH
    string run_date       [celé číslo] určuje zda je možný odpočet DPH
    string first_owner    [rm datum] rok [měsíc, den] konce STK
    string service_book   [celé číslo] rok [měsíc, den] od kdy je vozidlo v provozu
    string payment        [celé číslo] zda se jedná o prvního majitele vozidla
    string payment_count  [celé číslo] zda má vozidlo servisní knížku
    string note           [celé číslo] pokud je vůz na leasing je tu výše splátky v Kč
    string engine_power   [řetězec] pokud je vozidlo na leasing je tu počet splátek, které
    string url            [celé číslo] nutno uhradit
    string url            [řetězec] poznámka
    string url            [celé číslo] výkon motoru v kilowatttech
    string url            [řetězec][150] pokud si klient přeje aby se mu otevíral vlastní detail
    string url            [řetězec][150] inzerátu, je tu uložena url toho klientova inzerátu;
    string url            [řetězec][150] defaultně se otevírá detail na sautu
    string type_info      [řetězec][30] doplňující info o modelu
    string gas_mileage    [reálné číslo] průměrná spotřeba l / 100km
    struct kind_attributes{
        string motohodiny [celé číslo] atributy specifické pro druh vozidla
        string motohodiny [celé číslo] atribut uložen pouze pro pracovní stroje (kind_id=10)
    }
}
```

## Sauto: XML-RPC import

### Popis struct car\_data:

Popis:

Kromě car\_id jsou všechny položky v této struktuře typu string, ale ne všechny mohou být libovolným řetězcem. U každé položky je uvedeno jaký typ hodnot může obsahovat: [celé číslo] = celé číslo, [řetězec] = libovolný řetězec, [řetězec][n] = řetězec dlouhý maximálně n znaků, [rm datum] = datum ve formátu 'rrrr-mm-dd', 'rrrr-mm' nebo 'rrrr' (rrrr = rok, mm = měsíc, dd = den). Pokud vkládáme nový inzerát a nechceme nějakou položku vyplnit (tj. chceme ji nastavit hodnotu 'Neuvedeno'), tak ji buď do struktury car\_data vůbec neuvedeme nebo jí přiřadíme prázdný řetězec. Pokud inzerát editujeme a nechceme určitou položku měnit, tak ji do struktury neuvedeme, pokud chceme položce nastavit hodnotu 'Neuvedeno', tak jí přiřadíme prázdný řetězec.

Položka price\_leasing je nepovinná a nemusí být přítomna (je zachována zpětná kompatibilita).

Číselníky hodnot některých položek:

#### kind\_id

- číselník hodnot této položky je uveden v souboru [http://www.sauto.cz/car\\_list\\_xmlrpc.php](http://www.sauto.cz/car_list_xmlrpc.php).

#### manufacturer\_id

- číselník hodnot této položky je uveden v souboru [http://www.sauto.cz/car\\_list\\_xmlrpc.php](http://www.sauto.cz/car_list_xmlrpc.php).

#### model\_id

- číselník hodnot této položky je uveden v souboru [http://www.sauto.cz/car\\_list\\_xmlrpc.php](http://www.sauto.cz/car_list_xmlrpc.php).

#### body\_id

- číselník hodnot této položky je uveden v souboru [http://www.sauto.cz/car\\_list\\_xmlrpc.php](http://www.sauto.cz/car_list_xmlrpc.php).

#### state\_id

hodnota	význam
1	Česká Republika
2	Slovenská Republika
3	Francie
4	Itálie
5	Německo
6	Rakousko
7	Švýcarsko
8	Holandsko
9	Lucembursko
10	Jiná

#### fuel

hodnota	význam
1	benzín
2	nafta
3	LPG
4	elektro

Sauto: XML-RPC import

5	hybridní
6	CNG

tachometr\_unit

hodnota	význam
1	km
2	mil

condition

hodnota	význam
1	nové
2	ojeté
3	havarované
4	předváděcí
5	veterán
7	Nový (jen pro náhradní díly)
8	Použitý (jen pro náhradní díly)

dph

hodnota	význam
0	cena vozidla je bez DPH
1	cena vozidla je s DPH

vat\_deductable

hodnota	význam
0	odpočet DPH není možný
1	odpočet DPH je možný

first\_owner

hodnota	význam
0	není to první majitel
1	je to první majitel

service\_book

hodnota	význam
0	vozidlo nemá servisní knížku
1	vozidlo má servisní knížku

## Sauto: XML-RPC import

### Povinné položky:

V následující tabulce jsou uvedeny povinné položky (v car\_data) pro jednotlivé druhy (kind\_id) vozidel.

<b>kind_id</b>	<b>Povinné položky</b>
1 (osobní), 2 (terénní), 3 (motocykly), 4 (užitková), 5 (nákladní), 6 (autobusy), 9 (obytné), 11 (čtyřkolky)	kind_id
	manufacturer_id
	model_id
	body_id
	engine_volume
	made_date
	state_id
	tachometr
	tachometr_unit
	fuel
	color
	condition
	price
	dph
7 (přívěsy)	kind_id
	manufacturer_id
	model_id
	body_id
	made_date
	state_id
	color
	condition
	price
dph	
10 (pracovní stroje)	kind_id
	manufacturer_id
	model_id
	body_id
	made_date
	price
dph	

## Sauto: XML-RPC import

12 (náhradní díly)	kind_id
	manufacturer_id
	model_id
	body_id
	price
	condition

### Návratová hodnota:

```
struct{
    int status                status (200=OK,
                                404=Neplatne session_id,
                                405=Inzerát neexistuje,
                                406=Chyba v položkách (inzerátu),
                                407=Auto se zadanými kind_id, manufacturer_id, model_id
                                    a body_id v databázi neexistuje)
                                408=Vyčerpán počet modulů

    string status_message    slovní popis statusu

    struct output{
        int car_id           idéčko právě vloženého (zeditovaného) inzerátu (podle
                                číslování sauta)

        array error_items(   pole s položkami inz. ve kterých byla chyba
                                struct{
                                    string item     chybová položka
                                    int error        název položky, ve které byla chyba
                                                    kód chyby (1=Položka musí být vyplněna,
                                                            2=Chybně vyplněná položka,
                                                            3=Položka musí mít jedinečnou hodnotu, vyplněná
                                                                hodnota již v databázi existuje)
                                }
                                string error_message slovní popis chyby jaká v položce nastala
                                ...
        )
    }
}
```

### struct getCar(string session\_id, int car\_id)

- Vrátí informace z databáze pro dané auto.

Pro zadané idéčko auta vrátí všechny parametry, které bylo možné zadat pomocí metody addEditCar().

#### Parametry:

```
string session_id    cislo aktualni relace.
int car_id           id auta ktere se ma vratit
```

#### Návratová hodnota:

```
struct {
    int status        status (200=OK,
                            404=Neplatne session_id,
                            405=Inzerat neexistuje)

    string status_message    slovni popis statusu

    struct output {
        }                vraceny zaznam, struktura identicka
                            jako v metode addEditCar
    }
}
```

### struct getCarId(string session\_id, string custom\_id)

Sauto: XML-RPC import

### - **Vrátí car\_id pro zadané custom\_id.**

Pro zadané ídéčko inzerátu (vozidla) podle číslování klienta nalezne ídéčko inzerátu (vozidla) podle interního číslování serveru sauta, které se používá v některých dalších metodách.

#### **Parametry:**

```
string session_id  id session získané z metody getHash()
string custom_id   ídéčko inzerátu (vozidla) podle číslování klienta
```

#### **Návratová hodnota:**

```
struct{
    int status                status (200=OK,
                              404=Neplatne session_id,
                              405=Inzerát neexistuje)
    string status_message    slovní popis statusu
    struct output{
        int car_id           ídéčko inzerátu (vozidla) podle číslování sauta
    }
}
```

## **struct delCar(string session\_id, int car\_id)**

### - **Smaže inzerát (vozidlo).**

Podle zadaného car\_id vymaže tato metoda inzerát (vozidlo) z databáze.

#### **Parametry:**

```
string session_id  id session získané z metody getHash()
int car_id         ídéčko inzerátu (vozidla) podle číslování sauta
```

#### **Návratová hodnota:**

```
struct{
    int status                status (200=OK,
                              404=Neplatne session_id,
                              405=Inzerát neexistuje)
    string status_message    slovní popis statusu
}
```

## **struct listOfCars(string session\_id, string all)**

### - **Výpis všech inzerátů (vozidel), které má klient v databázi.**

Tato metoda vypíše seznam inzerátů (vozidel), které má klient v databázi. Pro každé vozidlo vrátí následující atributy: car\_id, custom\_id, car\_status, kind\_id, manufacturer\_id a model\_id.

#### **Parametry:**

```
string session_id  id session získané z metody getHash()
string all         [řetězec] nepovinný parametr vrátí inzeráty včetně ručně zadaných pro all='all'
```

#### **Návratová hodnota:**

```
struct{
    int status                status (200=OK,
                              404=Neplatne session_id)
    string status_message    slovní popis statusu
    struct output{
        array list_of_cars(
            struct{
                int car_id           pole s jednotlivými inzeráty
                string custom_id     záznam jednoho inzerátu (vozidla)
                int car_status       ídéčko inz. (vozidla) podle číslování sauta
                int kind_id          ídéčko inz. (vozidla) podle číslování klienta
                int manufacturer_id  stav inzerátu (aktivní/neaktivní)
                int model_id         ídéčko druhu vozidla
            }
        )
    }
}
```

## Sauto: XML-RPC import

```
        int model_id      idéčko modelu vozidla
    }
    ...
)
}
}
```

### **struct listOfAllCars(string session\_id, int client\_id)**

**- Vypis všech aktivních inzerátů (vozidel), volitelně jen pro daného klienta, volitelně od data.**

Tato metoda vypíše seznam aktivních inzerátů (vozidel). Pokud je `client_id != 0`, omezí výsledky jen na inzeráty klienta se zadaným id. Volitelně zobrazí pouze novější než. Volání bez parametru `client_id` není veřejně přístupné.

#### **Parametry:**

<code>string session_id</code>	id session získané z metody <code>getHash()</code>
<code>int client_id</code>	id prodejce. pokud je 0, vyberou se vsichni.
<code>string date</code>	datum, zobrazí se inzeráty novější než (yyyy-mm-dd)
<code>int offset</code>	index první instance v návratové hodnotě
<code>int limit</code>	maximální počet instancí v návratové hodnotě

#### **Návratová hodnota:**

```
struct {
    int status      status (200=OK,
                        404=Neplatne session_id,
                        405=Inzerat neexistuje)
    string status_message  slovní popis statusu
    struct output {
        array list_of_cars {
            int car_id      id vraceneho auto
        }
        int resultSize      pocet polozek
    }
}
```

### **struct addEditPhoto(string session\_id, int car\_id, struct photo\_data)**

**- Přidání a editace fotografie inzerátu.**

Tato metoda přidá k inzerátu fotografii nebo upraví některé její vlastnosti. Při editaci nelze upravit obrázek, ale jen atributy jako je popisek atd. Pokud bude `photo_id` v `photo_data` kladné (nenulové) celé číslo, tak se provede editace fotografie se shodným `photo_id`, pokud bude `photo_id` nula, tak se provede vložení nové fotografie.

#### **Parametry:**

<code>string session_id</code>	id session získané z metody <code>getHash()</code>
<code>int car_id</code>	id inzerátu (vozidla) podle číslování sauta
<code>struct photo_data{</code>	data vkládané fotografie (jednotlivé položky a jejich hodnoty)
<code>int photo_id</code>	id fotografie podle číslování sauta
<code>string main</code>	[celé číslo] říká zda se jedná o hlavní fotku
<code>string alt</code>	[řetězec] popis k fotce
<code>string client_photo_id</code>	[řetězec] id fotografie podle číslování klienta
<code>base64 b64</code>	[base 64] fotografie ve formátu JPEG zakódovaná pomocí base64
<code>}</code>	

#### **Popis struct photo\_data:**

*Popis:*

## Sauto: XML-RPC import

Kromě photo\_id a b64 jsou všechny položky v této struktuře typu string, ale ne všechny mohou být libovolným řetězcem. U každé položky je uvedeno jaký typ hodnot může obsahovat: [celé číslo] = celé číslo, [řetězec] = libovolný řetězec. Pokud vkládáme novou fotografii a nechceme nějakou položku vyplnit, tak ji buď do struktury photo\_data vůbec nevedeme nebo jí přiřadíme prázdný řetězec. Pokud fotografii editujeme a nechceme určitou položku měnit tak ji do struktury nevedeme, pokud chceme položce nastavit defaultní (většinou prázdnou) hodnotu tak jí přiřadíme prázdný řetězec. Pokud bude při editaci uvedena položka b64, bude ignorována. Pokud nebude při vkládání fotografií k inzerátu, který žádné nemá, uvedena žádná fotografie jako hlavní bude jako hlavní nastavena první vložená fotografie. Jelikož inzerát může mít jen jednu hlavní fotografii, tak pokud vkládáme hlavní fotografii k inzerátu, který už nějakou má, tak se z té staré stane obyčejná a jako hlavní se nastaví ta námi vkládaná.

*Číselníky hodnot některých položek:*

main

hodnota	význam
0	není to hlavní fotografie
1	je to hlavní fotografie

### Návratová hodnota:

```
struct{
    int status                status (200=OK,
                              404=Neplatne session_id,
                              405=Inzerát (se zadaným car_id) neexistuje,
                              406=Chyba v položkách (fotografie))

    string status_message    slovní popis statusu

    struct output{
        int photo_id         idéčko právě vložené (zeditované) fotografie (podle číslování
                              sauta; pozn. Pri chybe ukladani bude 0)

        array error_items(   pole s položkami fotografie, ve kterých byla chyba
            struct{          chybová položka
                string item   název položky, ve které byla chyba
                int error     kód chyby (1=Položka musí být vyplněna,
                              2=Chybně vyplněná položka,
                              3=Položka musí mít jedinečnou hodnotu, vyplněná
                              hodnota již v databázi existuje
                              4=Chyba retezce base64)

                string error_message    slovní popis chyby jaká v položce nastala
            }
        }
    }
}
```

## struct delPhoto(string session\_id, int photo\_id)

### - Smaže fotografii.

Smaže fotografii podle zadaného photo\_id. Photo\_id je jedinečné nejen v rámci jednoho inzerátu (vozidla), ale v rámci celého serveru sauta, proto pro smazání fotky není potřeba car\_id.

### Parametry:

```
string session_id    id session získané z metody getHash()
int photo_id         idéčko fotografie podle číslování sauta
```

### Návratová hodnota:

```
struct{
    int status                status (200=OK,
                              404=Neplatné session_id,
```

## Sauto: XML-RPC import

```
string status_message      409=Fotografie (se zadaným photo_id) neexistuje)
                             slovní popis statusu
}
```

### **struct getPhotoId(string session\_id, int car\_id, string client\_photo\_id)**

**- Podle zadaného client\_photo\_id (id fotografie podle klienta) a car\_id vrátí id fotografie podle číslování serveru sauta.**

Client\_photo\_id není jedinečné v rámci celého serveru sauta, ale je jedinečné v rámci jednoho inzerátu (vozidla).

#### **Parametry:**

```
string session_id      id session získané z metody getHash()
int car_id              idéčko inzerátu (vozidla) podle číslování sauta
string client_photo_id id fotografie podle číslování klienta
```

#### **Návratová hodnota:**

```
struct{
    int status              status (200=OK,
                             404=Neplatné session_id,
                             405=Inzerát neexistuje,
                             409=Fotografie (se zadaným client_photo_id) neexistuje)
    string status_message  slovní popis statusu
    struct output{
        int photo_id      idéčko fotografie podle číslování sauta
    }
}
```

### **struct listOfPhotos(string session\_id)**

### **struct listOfPhotos(string session\_id, int car\_id)**

**- Vypis všech fotek pro určitý inzerát (vozidlo).**

Vrátí seznam všech fotek daného inzerátu. Pokud voláme jednoparametrové přetížení, vypíšou všechny se fotografie daného klienta.

#### **Parametry:**

```
string session_id      id session získané z metody getHash()
int car_id              idéčko inzerátu (vozidla) podle číslování sauta
```

#### **Návratová hodnota:**

```
struct{
    int status              status (200=OK,
                             404=Neplatne session_id,
                             405=Inzerát neexistuje)
    string status_message  slovní popis statusu
    struct output{
        array list_of_photos(
            struct{
                int photo_id      záznam jedné fotografie
                string alt        idéčko fotografie podle číslování sauta
                int main          popisek fotografie
                                (1=fotografie je hlavní,
                                 0=fotografie není hlavní)
                string client_photo_id idéčko fotografie podle číslování klienta
                string filename   Cesta k souboru na serveru. Když se k této cestě
                                připojí koncovka .jpg, dostaneme cestu k velkému
                                obrázku, když se připojí _pre.jpg, dostaneme cestu
                                k náhledovému obrázku, a když se připojí _middle.jpg,
                                dostaneme cestu k obrázku střední velikosti.
            }
        )
    }
}
```

## **struct addEquipment(string session\_id, int car\_id, array int equipment)**

### **- Nastavení výbavy vozidla.**

Tato metoda nejprve smaže všechnu výbavu u zadaného vozidla a potom mu nastaví tu výbavu, která je uvedena v equipment. Pokud chceme výbavu u vozidla jen smazat pošleme equipment jako prázdné pole.

### **Parametry:**

```
string session_id    id session získané z metody getHash()
int car_id           idéčko inzerátu (vozidla) podle číslování sauta
array int equipment( pole s id položek výbavy
    int equipment_id id položky výbavy
    int equipment_id id položky výbavy
    ...
)
```

### **Popis array int equipment:**

#### *Číselník výbavy:*

- číselník výbavy je uveden v souboru [http://www.sauto.cz/equipment\\_list\\_xmlrpc.php](http://www.sauto.cz/equipment_list_xmlrpc.php). V souboru [http://www.sauto.cz/car\\_list\\_xmlrpc.php](http://www.sauto.cz/car_list_xmlrpc.php) je uvedeno jaké položky výbavy patří k jednotlivým druhům vozidel (kind\_id).

### **Návratová hodnota:**

```
struct{
    int status                status (200=OK,
                              404=Neplatné session_id,
                              405=Inzerát neexistuje,
                              410=Chyba v položkách výbavy)
    string status_message    slovní popis statusu
    struct output{
        array error_equipment( pole s položkami fotografie, ve kterých byla chyba
            struct{             chybová položka
                int equipment_id id položky vybavení, ve které byla chyba
                int error        kód chyby (1=Duplicitní položka,
                                   2=Položka pro zadaný druh neexistuje)
            }
            string error_message slovní popis chyby jaká v položce vybavení nastala
        }
    }
}
```

## **struct listOfEquipment(string session\_id, int car\_id)**

### **- Výpis výbavy pro zadané auto.**

Metoda pro auto zadané pomocí car\_id vrátí všechny položky jeho výbavy.

### **Parametry:**

```
string session_id    id session získané z metody getHash()
int car_id           idéčko inzerátu (vozidla) podle číslování sauta
```

### **Návratová hodnota:**

```
struct{
    int status                status (200=OK,
                              404=Neplatné session_id,
                              405=Inzerát neexistuje)
    string status_message    slovní popis statusu
    struct output{
        array equipment(      pole s id položek výbavy
            int equipment_id  id položky výbavy
            int equipment_id  id položky výbavy
            ...
        )
    }
}
```

Sauto: XML-RPC import

```
}  
  }  
)
```

## 2. Popis rozhraní

---

### Kódování

Veškerá komunikace se serverem probíhá v kódování UTF-8.

### Přihlášení

Pro přihlášení se nejprve musí zavolat metoda `getHash`, které se předá ID, a ta vrátí hashovací klíč, který se použije dále pro metodu `login`. Metodě `login` sloužící pro přihlášení se předá zakódované heslo a softwarový klíč. Pokud přihlášení proběhne v pořádku, vrátí metoda `status` 200.

### Typický průběh komunikace

`getHash`

`login`

`listOfCars`

`delCar`

...

`addEditCar`

`addEditPhoto`

`addEquipment`

...

`logout`

### 3. Chybové hlášky

status	slovní popis statusu
200	OK
210	Odhlášení je OK
401	Neexistující klient
402	Neexistující klient nebo špatné heslo
403	Neplatný klíč softwaru
404	Neplatné <code>session_id</code>
405	Inzerát neexistuje

## Sauto: XML-RPC import

406	Chyba v položkách inzerátu nebo fotografie
407	Auto se zadanými kind_id, manufacturer_id, model_id a body_id v databázi neexistuje
408	Vyčerpán počet modulů
409	Fotografie neexistuje
410	Chyba v položkách výbavy

## 4.Příklady

---

### PHP - PEAR

```
<?
/*****
Přihlaseň
*****/

// načtení třídy z PEARu
require_once 'XML/RPC.php';

// připojení na server
$client = new XML_RPC_Client('', '', );

// vytvoření dotazu – username klienta
$params = array(new XML_RPC_Value('uzivatelske_jmeno', ''));
$msg = new XML_RPC_Message('getHash', $params);

// posláň dotazu na server
$response = $client->send($msg);

// načtení výsledku
$getHash = XML_RPC_decode($response->value());

// je dotaz je OK ?
if ($getHash['status'] == 200){
    // překopírování outputu do zvlášť. proměnné
    $output = $getHash['output'];

    // vytvoření dotazu pro login a posláň na server
    $params = array(new XML_RPC_Value($output['session_id']),
        new XML_RPC_Value(md5(md5('xxx').$output['hash_key'])), new XML_RPC_Value('xxx'));
    $msg = new XML_RPC_Message("login", $params);
    $response = $client->send($msg);

    // načtení výsledku
    $login = XML_RPC_decode($response->value());

    // pokud je status 200, je to OK
    if ($login['status'] == 200){
        //jsme zalogováni, můžem importovat
        :
        :
        :
    }else{
        echo "Chyba [$login[status]]: $login[status_message]\n";
    }
}else{
    echo "Chyba [$getHash[status]]: $getHash[status_message]\n";
}
```

Sauto: XML-RPC import

```
}  
?>
```