1. Wstęp
2. Adres usługi
3. Konfiguracja
4. Metody
5. Typy danych
6. Przykład wywołania metody przy użyciu php i biblioteki nusoap
7. Odpowiedź serwera

Wstęp

Usługa udostępniona dla klientów serwisu pakka.pl, dla szybkiej wysyłki paczek. Warunkiem korzystania z webserwisu, jest posiadanie aktywnego konta. Do prawidłowej wymiany informacji, są potrzebne dane logowania w systemie pakka.pl.

Adres usługi

Usługa umieszczona jest pod adresem http://www.pakka.pl/api_services/run_service?wsdl.
Usługa działa poprawnie tylko dla użytkowników, którzy posiadają środki pieniężne na koncie, w tym celu trzeba doładować konto za pomocą serwisu dotpay, wybierając opcję testową.

Konfiguracja

Serwis bazuje na SOAP 1.1, WSDL 1.1 and HTTP 1.0/1.1.
Do poprawnego działania wymiany danych, konieczne jest założenie konta w serwisie.
Konto musi być aktywne i posiadać środki pieniężne, ponieważ zamówienia są realizowane na zasadzie prepaid.

Metody

Dostępne metody w systemie

1. PakkaSoapService.sendPackages(authData, parcels, packageType, sender, receiver, packagesOptions)

Metoda do składania zamówienia przy użyciu konta prepaid. Do prawidłowego złożenia zlecenia, niezbędne jest uzupełnienie swojego konta o środki pieniężne

3. PakkaSoapService.calculatePrice(authData, definiedParcels, sender, receiver, packagesOptions)
Funkcja wylicza cenę za dostarczenie przesyłki
4. PakkaSoapService.storeOrder(authData, definiedParcels, sender, receiver, packagesOptions)
Funkcja zapisuje w panelu pakka.pl zamówienie jako szablon, które można później wykorzystać do wysyłki.
Funkcja zwraca id szablonu.

Do testów można wykorzystać metodę calculatePrice, gdyż przyjmuje ona te same parametry co metoda sendPackage. sendPackage jako jedyna metoda realizuje zlecenie i jednocześnie pobiera kwotę z konta za złożenie zlecenia.

Typy danych

authData - dane autoryzacyjne

nazwa	*typ*	*opis*
username	Str	login serwisu pakka, email podany przy rejestracji
password	Str	hasło podane przy rejestracji

parcels - lista paczek do wysyłki

nazwa	*typ*	*opis*
width	Int	szerokość w cm
height	Int	wysokość w cm
depth	Int	głębokość w cm
weight	Int	waga w kg
content	Str	zawartość paczki

packageType - typ paczki, na razie obsługiwane są tylko zwykłe przesyłki, docelowo będą dodane palety

nazwa	*typ*	*opis*
packageType	Str	przy wysyłaniu danych trzeba podać typ jako “paczka”

sender - nadawca paczki

nazwa	*typ*	*opis*
name	Str	Imię i Nazwisko
postCode	Str	kod pocztowy w formacie XX-XXX
city	Str	Miasto
street	Str	Ulica
houseNumber	Str	Numer domu

phone	Str	telefon
email	Str	Email

<u>receiver</u> - odbiorca paczki

nazwa	*typ*	*opis*
name	Str	Imię i Nazwisko
postCode	Str	kod pocztowy w formacie XX-XXX
city	Str	Miasto
street	Str	Ulica
houseNumber	Str	Numer domu
phone	Str	telefon
email	Str	Email

<u>packagesOptions</u> - Usługi dodatkowe.

Na razie dostępna jest tylko usługa COD (za pobraniem), parametr packagesOptions przyjmuje następujące wartości

- COD - za pobraniem, parametr

nazwa	*typ*	*opis*
name	Str	Przyjmuje wartość “COD”, za pobraniem
value	Str	Wartość pobrania, w złotówkach
param	Str	Numer konta bankowego

<u>definiedParcels</u> - typ paczek, które mają zdefiniowane domyślne wymiary, uzyskanie listy tych paczek odbywa się przez wywołanie metody PakkaSoapService.parcelTypes

nazwa	*typ*	*opis*
packageTypeId	Str	identyfikator typu paczki, do pobrania typów paczki służy metoda **PakkaSoapService.parcelTypes**
content	Str	zawartość paczki

==========

Przykład wywołania metody PakkaSoapService.parcelTypes

Metoda zwraca typy paczek jakie mogą być zamówione

```
$data_to_send = array(
        'authData'=>array(
                'username'=>'admin',
                'password'=>'admin'
        ),
        );

$result = $client->call('PakkaSoapService.parcelTypes', $data_to_send);
```

Odpowiedź serwera

Jeśli zautoryzacja i dane są poprawne, serwer zwróci tablicę z rodzajami paczek

nazwa	*typ*	*opis*
packageTypeId	Str	Typ paczki, wykorzystywany w innych funkcjaw webserwisu
name	Str	Nazwa typu paczki
maxWeight	Int	Maksymalna waga paczki

```
Array
(
   [0] => Array
      (
         [packageTypeId] => do5kg
         [name] => Do 5kg
         [maxWeight] => 5
      )

   [1] => Array
      (
         [packageTypeId] => do10kg
         [name] => Do 10kg
         [maxWeight] => 10
      )

   [2] => Array
      (
         [packageTypeId] => do20kg
         [name] => Do 20kg
         [maxWeight] => 20
      )

   [3] => Array
```

```
    (
        [packageTypeId] => do31kg
        [name] => Do 31,5kg
        [maxWeight] => 31
    )

)
```

==========

Przykład wywołania metody PakkaSoapService.sendPackages

Używając tej metody należy pamiętać, że:
- maksymalna waga jednej paczki w przesyłce: 31,5 kg
- maksymalna długość: 175 cm (najdłuższy bok)
- suma obwodu podstawy i wysokości opakowania nie może przekroczyć 250 cm

```
$data_to_send = array(
        'authData'=>array(
                'username'=>'admin',
                'password'=>'admin'
        ),
        'parcels'=>array(
                array(
                        'width'=>4,
                        'height'=>12,
                        'depth'=>10,
                        'weight'=>10,
                        'content'=>'Pralka i lodówa'
                ),
                array(
                        'width'=>12,
                        'height'=>3,
                        'depth'=>14,
                        'weight'=>5,
                        'content'=>'Mini telewizorek'
                )
        ),

        'packageType'=>'paczka',
```

```
        'sender'=>array(
                'name'=>'Jan Kowalski',
                'postCode'=>'31-620',
                'city'=>'Kraków',
                'street'=>'Os. Piastów',
                'houseNumber'=>'111/111',
                'phone'=>'500500500',
                'email'=>'nadawca@asdfasdf.pl'
        ),
        'receiver'=>array(
                'name'=>'Janina Kowalska',
                'postCode'=>'00-005',
                'city'=>'Warszawa',
                'street'=>'Rysia',
                'houseNumber'=>'222/222',
                'phone'=>'900900900',
                'email'=>'odbiorca@odbiorcaxyz.pl'
        ),
        'packagesOptions'=>array(
                array(
                        'name'=>'COD',
                        'value'=>'12',
                        'param'=>'99 9999 99999 99999 9999 ….'
                )
        )
);

$result = $client->call('PakkaSoapService.sendPackages', $data_to_send);
```

Odpowiedź serwera

Jeśli zamówienie przejdzie poprawnie, serwer zwróci następujące dane

nazwa	*typ*	*opis*
Provider	Str	Jaki kurier obsłuży paczkę
PakkaOrderId	Int	Numer identyfikacyjny zamówienia w serwisie pakka.pl
Price	Str	Cena za przesyłkę
Status	Str	Jeśli wszystko przebiegło prawidłowo, zostanie zwrócony tekst “OK”
document_1	Str	Dokument wygenerowany przez kuriera, zgodny z base64 (list przewozowy, protokół przekazania)

document_2	Str	Dokument wygenerowany przez kuriera, zgodny z base64, (list przewozowy, protokół przekazania)

Array
(
[Provider] => Dpd
[PakkaOrderId] => 543
[Price] => 20.3
[Status] => OK
[document_1]=>JVBERi0xLjQKMyAwIG9iago8PC9UeXBlIC9QYWdlCi9QYXJlbn.....
[document_2]=>JVBERi0xLjQKMyAwIG9iago8PC9UeXBlIC9QYWdlCi9QYXJlbn.....
)

Przykład wywołania metody PakkaSoapService.calculatePrice

Metoda wylicza cenę za przesyłkę, parametry są takie same jak w przypadku sendPackage, więc na podstawie tej metody można testować funkcje

```
$data_to_send = array(
	'authData'=>array(
		'username'=>'admin',
		'password'=>'admin'
	),
	'parcels'=>array(
		array(
			'width'=>4,
			'height'=>12,
			'depth'=>10,
			'weight'=>10,
			'content'=>'Pralka i lodówa'
		),
		array(
			'width'=>12,
			'height'=>3,
			'depth'=>14,
			'weight'=>5,
			'content'=>'Mini telewizorek'
		)
	),

	'packageType'=>'paczka',

	'sender'=>array(
		'name'=>'Jan Kowalski',
		'postCode'=>'31-620',
		'city'=>'Kraków',
		'street'=>'Os. Piastów',
		'houseNumber'=>'111/111',
```

```
                'phone'=>'500500500',
                'email'=>'nadawca@asdfasdf.pl'
        ),
        'receiver'=>array(
                'name'=>'Janina Kowalska',
                'postCode'=>'00-005',
                'city'=>'Warszawa',
                'street'=>'Rysia',
                'houseNumber'=>'222/222',
                'phone'=>'900900900',
                'email'=>'odbiorca@odbiorcaxyz.pl'
        ),
        'packagesOptions'=>array(
                array(
                        'name'=>'COD',
                        'value'=>'12',
                        'param'=>'99 9999 99999 99999 9999 ….'
                )
        )
);

$result = $client->call('PakkaSoapService.calculatePrice', $data_to_send);
```

Odpowiedź serwera

Jeśli zautoryzacja i dane są poprawne, serwer zwróci tablicę z rodzajami paczek

nazwa	*typ*	*opis*
price	Float	Cena za przesłanie paczki
provider	Str	Nazwa kuriera

```
Array
(
   [price] => 20.3
   [provider] => DpdProvider
)
```

==========

Przykład wywołania metody PakkaSoapService.storeOrder

Funkcja zapisuje szablon zlecenia, w panelu pakka.pl można później wielokrotnie wykorzystywać takie zamówienie. Funkcja zwraca id szablonu

```
$data_to_send = array(
```

```
	'authData'=>array(
		'username'=>'admin',
		'password'=>'admin'
	),
	'parcels'=>array(
		array(
			'width'=>4,
			'height'=>12,
			'depth'=>10,
			'weight'=>10,
			'content'=>'Pralka i lodówa'
		),
		array(
			'width'=>12,
			'height'=>3,
			'depth'=>14,
			'weight'=>5,
			'content'=>'Mini telewizorek'
		)
	),

	'packageType'=>'paczka', //paczka


	'sender'=>array(
		'name'=>'Jan Kowalski',
		'postCode'=>'31-620',
		'city'=>'Kraków',
		'street'=>'Os. Piastów',
		'houseNumber'=>'111/111',
		'phone'=>'500500500',
		'email'=>'nadawca@asdfasdf.pl'
	),
	'receiver'=>array(
		'name'=>'Janina Kowalska',
		'postCode'=>'00-005',
		'city'=>'Warszawa',
		'street'=>'Rysia',
		'houseNumber'=>'222/222',
		'phone'=>'900900900',
		'email'=>'odbiorca@odbiorcaxyz.pl'
	),
	'packagesOptions'=>array(

	)
);

$result = $client->call('PakkaSoapService.storeOrder', $data_to_send);
```

Odpowiedź serwera

Jeśli zautoryzacja i dane są poprawne, serwer zwróci tablicę z rodzajami paczek

nazwa	*typ*	*opis*
	Int	Funkcja zwraca id szablonu