SDK Moip-PHP

O Moip-PHP é uma biblioteca que implementa uma camada de abstração para geração do XML de instruções do Moip, permitindo que você integre aos serviços de API sem poluir seu código com várias linhas de XML. Um exemplo rápido:

```
include_once "autoload.inc.php";

    $moip = new Moip();
    $moip->setEnvironment('test');
    $moip->setCredential(array(
        'key' => 'ABABABABABABABABABABABABABABABABABABABAB',
        'token' => '01010101010101010101010101010101'
        ));

    $moip->setUniqueID('ABC1234');
    $moip->setValue('100.00');
    $moip->setReason('Teste do Moip-PHP');

    $moip->validate('Basic');

    print_r($moip->send());
```

O MoIP-PHP utiliza o padrão Fluent Interfaces, portanto, você pode fazer o exemplo acima da seguinte forma:

```
include_once "autoload.inc.php";

    $moip = new Moip();

    print_r($moip->setEnvironment('test')
            ->setCredential(array(
        'key' => 'ABABABABABABABABABABABABABABABABABABABAB',
        'token' => '01010101010101010101010101010101'
        ))->setUniqueID('ABC1234')
            ->setValue('100.00')
            ->setReason('Teste do Moip-PHP')
            ->validate('Basic')
            ->send());
```

Métodos disponíveis

Veja baixo relação e detalhes dos métodos disponíveis que você poderá utilizar com o Moip-PHP.

Moip()

Método construtor.

Moip()

```
$moip = new Moip();
```

setEnvironment()

Método que define o ambiente em qual o requisição será processada, “test” para definir que será em ambiente de testes Moip o Sandbox, a omissão desse método define que a requisição deverá ser processada em ambiente real, de produção Moip.

Importante: ao definir o ambiente certifique-se de que está utilizando a autenticação correspondente ao ambiente, no Moip cada ambiente possui suas própria chaves de autenticação API.

setEnvironment($environment)

$environment : String (‘test’)

```
$moip->setEnvironment('test');
```

setCredential()

O Moip requer que você se autentique para que seja possível processar requisições em sua API, para isso antes de realizar qualquer requisição você deverá informar ao Moip suas credenciais da API formados por um *TOKEN* e uma *KEY*.

O parâmetro $credencials é um array associativo contendo as chaves key e token (ex: array(‘key’=>’sua_key’,'token’=>’seu_token’)). Se você ainda não possui estes dados, veja como obtê-las através em sua conta Sandbox.

setCredential($credential)

$credential : Array(‘key’,'token’)

```
$moip->setCredential(array(
        'key' => 'SUA_KEY',
        'token' => 'SEU_TOKEN'
        ));
```

validate()

O método *validate()* irá realizar a validação dos dados obrigatórios para o tipo de instrução que você deseja processar, você pode optar por um dos dois níveis de validação disponíveis o 'Basic' e 'Identification'.

- **Basic** : Irá realizar a validação dos dados mínimos de para uma requisição XML ao Moip.
- **Identification** : Irá validar os dados necessários para se processar um XML com identificação Moip, usados geralmente para redirecionar o cliente já no segundo step da pagina de pagamento no checkout Moip ou usar o Moip Transaparente.

validate($validateType)

$validateType : String ('Basic' ou 'Identification')

```
$moip->validate('Identification');
```

setUniqueID()

O método *setUniqueID()* atribui valor a tag "<IdProprio>" no XML Moip.

- **<IdProprio>:** Seu identificador único de pedido, essa mesma informação será enviada para você em nossos serviços de notificações de alterações de status para que você possa tratar e atualizar o status de seu pedido.

setUniqueID($id)

$id : String

```
$moip->setUniqueID('ABC1234');
```

setValue()

O método *setValue()* atribui valor a tag "<Valor>" no XML Moip.

- **<Valor>:** Responsável por definir o valor que deverá ser pago.

setValue($value)

$value : Numeric

```
$moip->setValue('100.00');
```

setAdds()

O método *setAdds()* atribui valor a tag “<Acrescimo>” no XML Moip.

- **<Acrescimo>:** Responsável por definir o valor adicional que deverá ser pago.

setAdds($value)
$value : Numeric

```
$moip->setAdds('15.00');
```

setDeduct()

O método *setDeduct()* atribui valor a tag “<Deducao>” no XML Moip.

- **<Deducao>:** Responsável por definir o valor de desconto que será subtraído do total a ser pago.

setDeduct($value)
$value : Numeric

```
$moip->setDeduct('15.00');
```

setReason()

O método *setReason()* atribui valor a tag “<Razao>” no XML Moip.

- **<Razao>:** Responsável por definir o motivo do pagamento.
  - Este campo é sempre obrigatório em um instrução de pagamento.

setReason($value)
$value : String

```
$moip->setReason('Pagamento de teste do Moip-PHP');
```

setPayer()

O método *setPayer()* atribui valores ao nodo “<Pagador>” no XML Moip.

- **<Pagador>:** Nodo de informações de quem está realizando o pagamento.

- **name** : **<Nome>** : Nome completo do pagador
- **email** : **<Email>** : E-mail do pagador
- **payerId** : **<IdPagador>** : Identificados unico do pagador
- **identity** : **<Identidade>** : Identidade do pagador (CPF)
- **phone** : **<TelefoneCelular>** : Telefone de contato secundário do pagador
- **billingAddress** : **<EnderecoCobranca>** : Endereço do pagador
  - **address** : **<Logradouro>** : Logradouro do pagador, rua, av, estrada, etc.
  - **number** : **<Numero>** : Numero residencial do pagador
  - **complement** : **<Complemento>** : Complemento do endereço do pagador
  - **city** : **<Cidade>** : Cidade do endereço do pagador
  - **neighborhood** : **<Bairro>** : Bairro do endereço do pagador
  - **state** : **<Estado>** : Estado do endereço do pagador em formato ISO-CODE (UF)
  - **country** : **<Pais>** : Pais do pagador em formato ISO-CODE
  - **zipCode** : **<CEP>** : CEP de endereço
  - **phone** : **<TelefoneFixo>** : Telefone de contato do pagador

setPayer($value)

$value : Array ('name','email','payerId','identity', 'phone','billingAddress' => Array('address','number','complement','city','neighborhood','state','country','zipCode','phone'))

```
$moip->setPayer(array('name' => 'Nome Sobrenome',
        'email' => 'email@cliente.com.br',
        'payerId' => 'id_usuario',
        'billingAddress' => array('address' => 'Rua do Zézinho Coração',
            'number' => '45',
            'complement' => 'z',
            'city' => 'São Paulo',
            'neighborhood' => 'Palhaço Jão',
            'state' => 'SP',
            'country' => 'BRA',
            'zipCode' => '01230-000',
            'phone' => '(11)8888-8888')));
```

addPaymentWay()

O método *addPaymentWay()* atribui valor a tag “<FormaPagamento>” do nodo “<FormasPagamento>” no XML Moip.

<FormaPagamento>: Define quais as formas de pagamento que serão exibidas ao pagador no Checkout Moip.

- **billet** : Para disponibilizar a opção “Boleto Bancário” como forma de pagamento no checkout Moip.
- **financing** : Para disponibilizar a opção “Financiamento” como forma de pagamento no checkout Moip.
- **debit** : Para disponibilizar a opção “Debito em conta” como forma de pagamento no checkout Moip.
- **creditCard** : Para disponibilizar a opção “Cartão de Crédito” como forma de pagamento no checkout Moip.

- **debitCard** : Para disponibilizar a opção “Cartão de débito” como forma de pagamento no checkout Moip.

addPaymentWay($way)

$way : String ('billet','financing','debit','creditCard','debitCard')

```
$moip->addPaymentWay('creditCard');
$moip->addPaymentWay('billet');
$moip->addPaymentWay('financing');
$moip->addPaymentWay('debit');
$moip->addPaymentWay('debitCard');
```

setBilletConf()

O método *setBilletConf()* atribui valores ao node “<Boleto>” no XML Moip que é responsável por definir as configurações adicionais e personalização do Boleto bancário.

- **$expiration :** Data em formato “AAAA-MM-DD” ou quantidade de dias.
- **$workingDays :** Caso “$expiration” seja quantidade de dias você pode definir com “true” para que seja contado em dias úteis, o padrão será dias corridos.
- **$instructions :** Mensagem adicionais a ser impresso no boleto, até três mensagens.
- **$uriLogo :** URL de sua logomarca, dimensões máximas 75px largura por 40px altura.

setBilletConf($expiration, $workingDays, $instructions, $uriLogo)

$expiration : Int ou Date
$workingDays : Boolean
$instructions : Array()
$uriLogo : String

```
$moip->setBilletConf("2011-04-06",
            false,
            array("Primeira linha",
                "Segunda linha",
                "Terceira linha"),
            "http://seusite.com.br/logo.gif");
```

addMessage()

O método *addMessage()* atribui valor a tag “<Mensagem>” do node “<Mensagens>” no XML Moip.

- **<Mensagens>:** Node com “<Mensagens>”.
  - **<Mensagem>:** TAG que define mensagem adicional a ser exibida no checkout Moip.

addMessage($msg)

$msg : String

```
$moip->addMessage('Seu pedido contem os produtos X,Y e Z.');
```

setReturnURL()

O método *setReturnURL()* atribui valor a tag “<URLRetorno>” no XML Moip, responsável por definir a URL que o comprador será redirecionado ao finalizar um pagamento através do checkout Moip.

setReturnURL($url)

$url : String

```
$moip->setReturnURL('https://meusite.com.br/cliente/pedido/bemvindodevolta');
```

setNotificationURL()

O método *setNotificationURL()* atribui valor a tag “<URLNotificacao>” no XML Moip, responsável por definir a URL ao qual o Moip deverá notificar com o NASP (Notificação de Alteração de Status de Pagamento) as mudança de status.

setNotificationURL($url)

$url : String

```
$moip->setNotificationURL('https://meusite.com.br/nasp/');
```

addComission()

O método *addComission()* atribui valores as tags “<Comissoes>” no XML Moip, responsável por atribuir recebedores secundários a transação.

- **$reason** : Razão/Motivo ao qual o recebedor secundário receberá o valor definido.
- **$receiver**: Login Moip do usuario que receberá o valor.
- **$value** : Valor ao qual será destinado ao recebedor secundário.
- **$percentageValue**: Caso “true” define que valor será calculado em relação ao percentual sobre o valor total da transação.
- **$ratePayer**: Caso “true” define que esse recebedor secundário irá pagar a Taxa Moip com o valor recebido.

addComission($reason, $receiver, $value, $percentageValue, $ratePayer)

$reason : String
$receiver : String
$value : Number
$percentageValue: Boolean
$ratePayer : Boolean

```
$moip->addComission('Razão do Split', 'recebedor_secundario', '5.00');
$moip->addComission('Razão do Split', 'recebedor_secundario_2', '12.00', true, true);
```

addParcel()

O método *addParcel()* atribui valores as tags de "<Parcelamentos>" no XML Moip, responsável configuras as opções de parcelamento que serão disponíveis ao pagador.

- **$min** : Quantidade mínima de parcelas disponível ao pagador.
- **$max** : Quantidade máxima de parcelas disponíveis ao pagador.
- **$rate** : Valor de juros a.m por parcela.
- **$transfer** : Caso "true" define que o valor de juros padrão do Moip será pago pelo pagador.

addParcel($min, $max, $rate, $transfer)

$min : Number
$max : Number
$rate : Number
$transfer : Boolean

```
$moip->addParcel('2', '4');
$moip->addParcel('5', '7', '1.00');
$moip->addParcel('8', '12', null, true);
```

setReceiver()

O método *setReceiver()* atribui valor a tag "<LoginMoIP>" do node "<Recebedor>" que identifica o usuário Moip que irá receber o pagamento no Moip.

- **$receiver** : Login Moip do recebedor primario.

setReceiver($receiver)

$receiver : String

```
$moip->setReceiver('integracao@labs.moip.com.br');
```

getXML()

O método *getXML()* irá retornar o XML gerado com todos os atributos que você configurou, esse método pode ajudar a saber exatamente o XML que você irá enviar ao Moip.

getXML()

```
$moip = new Moip();
$moip->setEnvironment('test');
$moip->setCredential(array(
    'key' => 'ABABABABABABABABABABABABABABABABABABABAB',
    'token' => '01010101010101010101010101010101'
    ));
$moip->setUniqueID('ABC1234');
$moip->setValue('100.00');
$moip->setReason('Teste do Moip-PHP');
$moip->validate('Basic');

print_r($moip->getXML());


//IRÁ IMPRIMIR
<?xml version="1.0" encoding="utf-8"?>
<EnviarInstrucao>
    <InstrucaoUnica>
        <IdProprio></IdProprio>
        <Razao>Teste do Moip-PHP</Razao>
        <Valores>
            <Valor moeda="BRL">100.00</Valor>
        </Valores>
    </InstrucaoUnica>
</EnviarInstrucao>
```

send()

O método *send()* executa o envio da instrução ao Moip, e retorna os dados de resposta obtidos do Moip.

- **response** : “true” para o caso de sucesso e “false” para quando ocorre algum erro.
- **error** : Retorna sempre uma mensagem quando “response” é “false”.
- **xml**: Retorna sempre o XML de resposta Moip quando “response” é “true”.

.

send()

```
$moip = new Moip();
$moip->setEnvironment('test');
```

```
$moip->setCredential(array(
    'key' => 'ABABABABABABABABABABABABABABABABABABABAB',
    'token' => '01010101010101010101010101010101'
    ));
$moip->setUniqueID('ABC1234');
$moip->setValue('100.00');
$moip->setReason('Teste do Moip-PHP');
$moip->validate('Basic');

print_r($moip->send());


//IRÁ IMPRIMIR
stdClass Object
(
    [response] => 1
    [error] =>
[xml] => <ns1:EnviarInstrucaoUnicaResponse xmlns:ns1="http://www.moip.com.br/ws/
alpha/"><Resposta><ID>201209042007216380000000989104</ID><Status>Sucesso</
Status><Token>M2C031R2Q0Z9W0Y4Q2S0H0W7E2G1Z6P3E8C0C0W050T01070Y9Y8V9G1F0F4</Token></Resposta></
ns1:EnviarInstrucaoUnicaResponse>
)
```

getAnswer()

O método *getAnswer()* retorna os dados de resposta do Moip em forma de objeto.

- **response** : “true” para o caso onde o “<Status>” Moip retornou “Sucesso” e “false” para quando retornou “Falha”.
- **error** : Retorna sempre uma mensagem quando “response” é “false”.
- **token**: Retorna o TOKEN de pagamento gerado para quando “response” é “true”.
- **payment_url** : Retorna a URL de checkout Moip preparada para redirecionar o cliente com o TOKEN de pagamento para quando “response” é “true”.

.

getAnswer()

```
$moip = new Moip();
$moip->setEnvironment('test');
$moip->setCredential(array(
    'key' => 'ABABABABABABABABABABABABABABABABABABABAB',
    'token' => '01010101010101010101010101010101'
    ));

$moip->setUniqueID('ABC1234');
$moip->setValue('100.00');
$moip->setReason('Teste do Moip-PHP');
$moip->validate('Basic');
$moip->send();
```

```
print_r($moip->getAnswer());

//IRÁ IMPRIMIR
stdClass Object
(
    [response] => 1
    [error] =>
    [token] => 92D091R2I0Y9X0E4T2K034L2H2V4H2J6L9R0S0T0K0N0L0T0Y9H879H144O8
[payment_url] => https://desenvolvedor.moip.com.br/sandbox/Instrucao.do?
token=92D091R2I0Y9X0E4T2K034L2H2V4H2J6L9R0S0T0K0N0L0T0Y9H879H144O8
)
```

queryParcel()

O método *queryParcel()* retorna um Array() contendo as informações de parcelas e seus respectivos valores cobrados por parcela e o valor total a ser pago referente a taxa de juros simulada..

- **REQUEST**
  - **$login**: Login Moip do usuario.
  - **$maxParcel**: Máximo de parcelar a ser consultado.
  - **$rate**: Taxa de juros para simulação.
  - **$simulatedValue**: Valor pago ao qual será simulado.

- **RESPONSE**
  - **response** : “true” em caso de resposta Moip com “<Status>” “Sucesso” e “false” em caso de “Falha”
  - **installment**: Numero de parcela correspondente aos valores.
    - **total** : Total a ser pago.
    - **rate**: Taxa de juros atribuido.
    - **value**: Valor por parcela.

queryParcel($login, $maxParcel, $rate, $simulatedValue)

$login : String
$maxParcel : Number
$rate : Number
$simulatedValue: Number

```
$moip = new Moip();
$moip->setEnvironment('test');
$moip->setCredential(array(
    'key' => 'ABABABABABABABABABABABABABABABABABABABAB',
    'token' => '01010101010101010101010101010101'
    ));

print_r($moip->queryParcel('integracao@labs.moip.com.br', '4', '1.99', '100.00'));

//IRÁ IMPRIMIR
```

```
Array
(
    [response] => 1
    [installment] => Array
        (
            [1] => Array
                (
                    [total] => 100.00
                    [rate] => 1.99
                    [value] => 100.00
                )

            [2] => Array
                (
                    [total] => 103.00
                    [rate] => 1.99
                    [value] => 51.50
                )

            [3] => Array
                (
                    [total] => 104.01
                    [rate] => 1.99
                    [value] => 34.67
                )

            [4] => Array
                (
                    [total] => 105.04
                    [rate] => 1.99
                    [value] => 26.26
                )

        )

)
```