Manual de integração
Via Webservices

ÍNDICE

1. ACOMPANHAMENTO DAS MODIFICAÇÕES

2. APRESENTAÇÃO

Este documento apresenta os padrões de webservices para automatizar as ações manuais realizadas pelo comerciante na ferramenta de gestão do PayZEN, o Backoffice.

Estes webservices foram desenvolvidos com base em SOAP (Simple Object Access Protocol) e são descritos pelo arquivo wsdl disponível em: https://secure.payzen.com.br/vads-ws/v3?wsdl

Para garantir uma interação segura, os webservices (SOAP) são criptografados usando o protocolo HTTPS. Além disso, um mecanismo de assinatura foi criado para validar e autenticar a troca de dados.

Antes de começar sua integração, por favor, certifique-se que seu plano de serviços do PayZEN inclui o uso de webservices

3. DESCRIÇÃO DOS TIPOS

Além dos tipos simples (int, string, date, etc.), os webservices utilizam tipos mais complexos (código de retorno, descrição da transação, etc.) que serão descritos neste capitulo.

Para os campos de datas devem se seguir as recomendações do W3C (http://www.w3.org/TR/NOTE-datetime). Por exemplo uma data de validade em Dezembro de 2011, o formato correto deve ser 2011-12-31T23:59:59+01:00.

3.1. standardResponse

Este tipo é usado para descrever a resposta maioria dos webservices

Nome do campo	Tipo	Descrição
errorCode	Int	Código de erro (cf. 2.6)
extendedErrorCode	String	Precisão do código de erro
transactionStatus	Int	Status da transação
timestamp	Long	Timestamp para geração de assinatura única
signature	String	Assinatura da transação (cf.4)

Os diferentes valores dos códigos de erros são específicos para cada webservice

Os diferentes status da transação podem ser:

Valor	Descrição
0	Inicial (em tratamento)
1	A validar
2	A forçar – Contatar o emissor
3	A validar e autorizar
4	Pendente captura
5	Pendente autorização
6	Capturado
7	Expirado
8	Recusado
9	Cancelado
10	Pendente
11	Captura em andamento
12	Autorização em andamento
13	Falha

A assinatura permite validar a integridade da resposta. O calculo da assinatura é feito com os parâmetros na seguinte ordem:

errorCode, extendedErrorCode, transactionStatus, timestamp

3.2. localControl

Este tipo é usado para descrever um controle local

Nome do campo	Tipo	Descrição
name	String	Nome do controle
Result	Bool	Resultado do controle

Os diferentes valores possíveis para o campo “name” são:

Valor	Descrição
“CARD”	Cartão registrado numa lista cinza
“COUNTRY”	Países registrados numa lista cinza ou ausentes numa lista branca
“IPADDR”	Endereço IP registrado numa lista cinza
“AMOUNT”	Valor excepcional


Esta lista é suscetível de aumentar, favor considerar isso em seu desenvolvimento.

3.3. transactionInfo

Este tipo permite descreve uma transação

NOME DO CAMPO	TIPO	DESCRIÇÃO
Resposta geral		
erroCode	Int	Código de erro (cf.2.6)
extendedErrorCode	String	Precisão do código de erro
transactionStatus	Int	Status da transação (cf.2.1 para detalhes)
Detalhe da transação		
shopId	String	Identificação da Loja
PaymentMethod	String	Canal de pagamento (“VPC”;”E-COMMERCE”;”CALLCENTER”)
contractNumber	String	Número do contrato o comerciante
orderId	String	Número do pedido
orderInfo	String	Descrição livre do pedido
orderInfo2	String	Descrição livre do pedido
orderInfo3	String	Descrição livre do pedido
transmissionDate	Date	Data da transação
transactionId	String	Identificação da transação
sequenceNb	Int	Numero sequencial da transação
amount	Long	Valor total da transação incluindo casa decimal
initialAmount	Long	Valor inicial (antes de alteração) incluindo casa decimal
Devise	Int	Moeda (código da moeda ISSO 4217, Real 986)
cvAmount	Long	Montante em contra valor incluindo casa decimal
cvDevise	Int	Moeda em contra valor (código da moeda ISSO 4217, Real 986)
presetationDate	Date	Data da captura do pedido
type	Int	0 = DÉBITO; 1 = CRÉDITO
multiplePaiement	Int	Pagamento em N vezes (Não = 0; Sim = 1)
ctxMode	String	Ambiente plataforma de pagamento (“TEXT”; ”PRODUCTION”)
Detalhes do cartão		
cardNumber	String	Número do cartão

cardNetwork	String	Rede de adquirente
carType	String	Tipo do cartão
cardCountry	Int	País de emissor(Código numérico ISO 31661-1. Ex. France=250)
cardExpirationDate	Date	Data de validade do cartão
Detalhes do portador		
customerId	String	Código do cliente
customerTitle	String	Tratamento
customerName	String	Nome do cliente
customerPhone	String	Telefone do cliente
customerMail	String	E-mail do cliente
customerAddress	String	Endereço do cliente
customerZipCode	String	Código postal do cliente (CEP)
customerCity	String	Cidade do cliente
customerCountry	String	País do cliente
customerLanguage	String	Idioma do cliente (Código ISSO 639-1, com 2 caracteres)
customerIP	String	Endereço IP do cliente
Detalhes da autenticação 3D-Secure		
transactionCondition	String	"3D_SUCCESS";"3D_FAILURE";"3D_ERROR"; "3D_NOTENROLLED";"3D_ATTEMPT";"SSL" (detalhes abaixo)
vadsEnrolled	String	Registro do portador 3DS
vadsStatus	String	Autenticação do portador
VadsECI	String	Indicador de comércio eletrônico
vadsXID		Identificador de transação 3DS
vadsCAVV	String	Informações relativas ao tratamento do criptograma de comércio eletrônico
vadsCAVVAlgorithm	String	Método de calculo do criptograma de comércio eletrônico
vadsSignatureValid	String	Assinatura de autenticação
directoryServer	String	Rede onde o Directory Server foi conectado
Detalhes da autorização		
authMode	String	"MARK" com impressão e "FULL" sem impressão
markAmount	Long	Valor impresso incluindo casa decimal
markDevise	Int	Moeda impressa (código da moeda ISSO 4217, Real 986)
markDate	Date	Data impressa
markNb	String	Número da autorização impressa
markResult	Int	Resultado impresso
markCVV2_CVC2	String	Informação relativa ao tratamento do criptograma visual impresso
authAmount	Long	Montante da autorização incluindo casa decimal
authDevise	Int	Moeda da autorização (código da moeda ISSO 4217, Real 986)
authDate	Date	Data da autorização
authNb	String	Número da autorização
authResult	Int	Resultado da autorização
authCVV_ CVC2		Informação relativa ao tratamento do criptograma da autorização
Detalhes da garantia & Controles locais		
warrantlyResult	String	Garantia de pagamento
LocalControl	Array	Tabela dos resultados de vários controles locais
Detalhes da captura (especificado somente se a transação foi capturada)		
captureDate	Date	Data da captura
captureNumber	Int	Número da captura
rapprochementStatus	Int	Status da conciliação bancária
refundAmount	Long	Valor sujeito a reembolso incluindo casa decimal
refundDevise	Int	Moeda do reembolso (Código moeda ISO 4217, Real : 986)
litige	Bool	Litígio
timestamp	long	Timestamp para geração da assinatura única

signature	String	Assinatura da transação (cf. 4)

Detalhes do parâmetro transactionCondition

Valor	Descrição
"3D_SUCCESS"	O comerciante e o comprador estão registrados em 3-D Secure e o comprador foi autenticado com êxito.
"3D_FAILURE"	O comerciante e o comprador estão registrados em 3-D Secure mas o comprador não conseguiu autenticar (senha errada)
"3D_ERROR"	O comerciante participa do 3-D Secure, mas o PayZEN encontrou um problema técnico durante o processo de autenticação (quando a verificação do cartão de inscrição no programa 3D ou autenticação do comprador)
"3D_NOTENROLLED"	O comerciante participa do 3-D Secure, mas o cartão do comprador não está inscrito
"3D_ATTEMPT"	O comerciante e o comprador estão inscritos no 3-D Secure, mas o comprador não tem que ser autenticado (o controle de acesso ao servidor do banco que emitiu o cartão que implementa a geração de uma prova de tentativa de autenticação).
"SSL"	O comerciante não está inscrito no 3D-Secure ou o canal de vendas está coberto por esta garantia.

A assinatura valida a integridade da resposta, o cálculo desta assinatura é feito tendo os parâmetros na seguinte ordem:

errorCode, extendedErrorCode, transactionStatus, shopId, paymentMethod, contractNumber, orderId, orderInfo, orderInfo2, orderInfo3, transmissionDate, transactionId, sequenceNb, amount, initialAmount, devise, cvAmount, cvDevise, presentationDate, type, multiplePaiement, ctxMode, cardNumber, cardNetwork, cardType, cardCountry, cardExpirationDate, customerId, customerTitle, customerName, customerPhone, customerMail, customerAddress, customerZipCode, customerCity, customerCountry, customerLanguage, customerIP, transactionCondition, vadsEnrolled, vadsStatus, vadsECI, vadsXID, vadsCAVVAlgorithm, vadsCAVV, vadsSignatureValid, directoryServer, authMode, markAmount, markDevise, markDate, markNb, markResult, markCVV2_CVC2, authAmount, authDevise, authDate, authNb, authResult, authCVV2_CVC2, warrantlyResult, captureDate, captureNumber, rapprochementStatut, refoundAmount, refundDevise, litige, timestamp

3.4. createPaiementInfo

Este tipo é usado para descrever os parâmetros na criação de uma transação

Nome do campo	Tipo	Descrição	Obrigatório
Detalhes da transação			
shopId	String	Identificação da loja	X
transmissionDate	Date	Data da transação	X
transactionId	String	Identificação da transação	X
PaymentMethod	String	EC=E-commerce;BO=Backoffice;MOTO=Email ou telefone; CC=Call center; OTHER=Outros	X
orderId	String	Código do pedido	X
orderInfo	String	Descrição livre do pedido	
orderInfo2	String	Descrição livre do pedido	
orderInfo3	String	Descrição livre do pedido	
amount	Long	Valor da transação com casa decimal	X
devise	int	Moeda (código ISO 4217, Real 986)	X
presentationDate	Date	Data de captura	
validationMode	int	0 = Automático; 1 = Manual	
Detalhes do cartão			
cardNumber	String	Número do cartão	X
cardNetwork	String	Rede adquirente(REDECARD_VISA; REDECARD_MASTERCARD; CIELO_VISA; CIELO_DINERS; CIELO_MASTERCARD)	X
cardExpirationDate	Date	Data de validade do cartão	X
Cvv	String	Criptograma visual	
contractNumber	String	Número do contrato do comerciante	
Parâmetros avançados			
ThreeDsResult	ThreeDsResult	Cf 2.5	
SubPaymentType	Integer	Deixar em branco	
SubReference	String	Deixar em branco	
SubPaymentNumber	Integer	Deixar em branco	
Detalhes do portador			
customerId	String	Código do cliente	
customerTitle	String	Tratamento (Sr., Sra. Srta.)	
customerName	String	Nome do cliente	
customerPhone	String	Telefone do cliente	
customerMail	String	E-mail do cliente	
customerAddress	String	Endereço do clienyte	
customerZipCode	String	Código postal - CEP	
customerCity	String	Cidade do cliente	
customerCountry	String	País do cliente	
customerLanguage	String	Idioma (Code ISO 639-1, 2 caracteres)	
customerIP	String	Endereço IP do cliente	
customerSendMail	Bool	Envio de e-mail ao cliente	
Diversos			
ctxMode	String	Ambiente de pagamento (“TEXT”; ”PRODUCTION”)	X
comment		Comentário livre	

3.5. ThreeDsResult

Este tipo permite descrever os parâmetros de retorno 3DS

Nome do campo	Tipo	Descrição	Obrigatório
Brand	String	Bandeira do cartão (VISA ou MASTERCARD)	X
Enrolled	String	Status da inscrição do portador • Y : Inscrito • N : Não inscrito • U : Desconhecido	X
authStatus	String	Status autenticação Y : Autenticado 3DS N : Erro na autenticação U : Autenticação impossível A : Autenticação de teste	X
Eci	String	ECI	
Xid	String	XID	X
cavv	String	CAVV	
cavvAlgorithm	String	Algoritimo CAVV • 0 : HMAC • 1 : CVV • 2 : CVV_ATN • 3 : Mastercard SPA	

3.6. errorCode

ErrorCode	Tipo
0	Ação realizada com sucesso
1	Ação não autorizada
2	Transação não encontrada
3	Transação não em boas condições
4	Transação já existe
5	Assinatura errada
6	Data errada
10	Valor errado
11	Moeda errado
12	Tipo de cartão desconhecido
13	Parâmetro "data de validade" inválido
14	Parâmetro "cvv" inválido
15	Contrato desconhecido
16	Parâmetro "Número do cartão" inválido
17	Identificação não encontrada
18	Identificação não é valida (Terminado, …)
19	Inscrição não encontrada
20	Inscrição invalida
21	Identificação já existente
22	Criação de identificação recusada
40	Faixa não encontrada
50	Parâmetro “shopId” inválido
51	Parâmetro “tramissionDate” inválido
52	Parâmetro “transactionId” inválido
53	Parâmetro “ctxMode” inválido

54	Parâmetro “comment” inválido
55	Parâmetro “AutoNb” inválido
56	Parâmetro ”AutoDate” inválido
57	Parâmetro “presentationDate” inválido
58	Parâmetro “newTransactionId” inválido
59	Parâmetro “validationMode” inválido
60	Parâmetro “orderId” inválido
61	Parâmetro “orderInfo1” inválido
62	Parâmetro "orderInfo2" inválido
63	Parâmetro "orderInfo3" inválido
64	Parâmetro "paymentMethod"inválido
65	Parâmetro "CardNumber" inválido
66	Parâmetro "ContractNumber"inválido
67	Parâmetro "customerId" inválido
68	Parâmetro "customerTitle" inválido
69	Parâmetro "customerName"inválido
70	Parâmetro "customerPhone"inválido
71	Parâmetro "customerMail" inválido
72	Parâmetro "customerAddress"inválido
73	Parâmetro "customerZipCode"inválido
74	Parâmetro "customerCity" inválido
75	Parâmetro "customerCountry"inválido
76	Parâmetro "customerLanguage" inválido
77	Parâmetro "customerIp" inválido
78	Parâmetro "customerSendMail" inválido
79	Parâmetro ”customerMobilePhone” inválido
80	Parâmetro "subPaiementType" inválido
81	Parâmetro "subReference" inválido
82	Parâmetro "initialAmount" inválido
83	Parâmetro "occInitialAMount" inválido
84	Parâmetro "effectDate" inválido
85	Parâmetro "state" inválido
90	Parâmetro "enrolled" inválido
91	Parâmetro "authStatus " inválido
92	Parâmetro "eci" inválido
93	Parâmetro "xid" inválido
94	Parâmetro "cavv" inválido
95	Parâmetro "cavvAlgo" inválido
96	Parâmetro "brand" inválido
93	Outro erro

Precisão dos códigos de erros

ErrorCode 0: Indica que a ação solicitada foi realizada com sucesso, indicando que o formato do pedido está correto.

Observação: No caso da criação de um pagamento (método criar), este código de erro não deve ser confundido com o campo TransactionStatus, utilizado unicamente para informar o resultado do pagamento. Assim, podemos ter um código de erro de 0 e um TransactionStatus 8, correspondente à criação de uma transacção cujo pedido de autorização foi negado.

ErrorCode 1: Indica que seu plano de serviços do PayZEN não possui acesso aos webservices, contate nosso suporte.

4. DESCRIÇÃO DO MÉTODO

4.1. Cancel

Esta função permite cancelar definitivamente uma transação, ainda não descontada, com um dos status seguintes:

- A validar
- A validar e autorizar
- Em espera
- Em espera d'auto
- Em espera da captura

Esta função leva em consideração os parâmetros seguintes:

Nome do campo	Tipo	Descrição	Obrigatório
shopId	String	Identificador da loja	
transmissionDate	Date	Data da transação	
transactionId	String	Identifcador da transação	
sequenceNb	Int	Número de sequência da transação	
ctxMode	String	Contexto do ambiente ("TEST","PRODUCTION")	
Comment	String	Comentário "livre"	
wsSignature	String	Assinatura (cf §)	

O cálculo da assinatura faz-se tomando os parâmetros na ordem seguinte: **shopId, transmissionDate, transactionId, sequenceNb, ctxMode, comment**

Essa função retorna uma resposta do tipo StandardResponse (cf 3.1). Os códigos de erros (errorCode) possiveis são:

errorCode	Descrição
0	Ação realizada com sucesso
1	Ação não autorizada
2	Transação não encontrada
3	Transação não colocada no Status correto
5	Assinatura errada
50	Parâmetro 'shopId' Inválido
51	Parâmetro 'transmissionDate' Inválido
52	Parâmetro 'transactionId' Inválido
53	Parâmetro 'ctxMode' Inválido
54	Parâmetro 'comment' Inválido
99	Erro desconhecido

NB: O código de erro extenso (extendedErrorCode) não é informado para esta função.

4.2. Validate

Esta função permite autorizar a captura no banco de uma transação na data de apresentação pedida no pagamento original. As transações que podem são sujeitas à validação possuem um dos status seguintes:

- Para validar
- Para validar e autorizar

função leva em consideração os parâmetros seguintes:

Nome do campo	Tipo	Descrição	Obrigatório
shopId	String	Identificador da loja	shopId
transmissionDate	Date	Data da transação	transmissionDate
transactionId	String	Identificador da transação	transactionId
sequenceNb	Int	Número SEQUENCIALda transação	sequenceNb
ctxMode	String	Contexto do ambiente ("TEST", "PRODUCTION")	ctxMode
comment	String	Comentário "livre"	comment
WsSignature	String	Assinatura(cf §5)	WsSignature

Esta função retorna uma resposta do tipo Standard Response (cf 3.1).

Os códigos de erros (errorCode) possíveis são:

errorCode	Descrição
0	Ação realizada com sucesso
1	Ação não autorizada
2	Transação não encontrada
3	Transação não colocada no Status correto
5	Assinatura errada
50	Parâmetro ‘shopId’ Inválido
51	Parâmetro ‘transmissionDate’ Inválido
52	Parâmetro ‘transactionId’ Inválido
53	Parâmetro ‘ctxMode’ Inválido
54	Parâmetro ‘comment’ Inválido
99	Erro desconhecido

NB: O código de erro extenso (extendedErrorCode) não está informado para esta função.

4.3. Force

Esta função permite ao comerciante transmitir o número da autorização de uma transação em seguida a uma chamada de voz.

Só as transações do Status "Para forçar" podem beneficiar desta função.

Esta função leva em consideração os seguintes parâmetros:

Nome do campo	Tipo	Descrição	Obrigatório
shopId	String	Identificador da loja	X
transmissionDate	Date	Data da transação	X
transactionId	String	Identificador da transação	X
sequenceNb	Int	Número de sequência da transação	X
ctxMode	String	Contexto do ambiente da plataforma ("TEST", "PRODUCTION")	X
autorisationNb	String	Número de autorização	X
autorisationDate	Date	Data de autorização	X
comment	String	Comentário "livre"	
WsSignature	String	Assinatura (cf §5)	X

O cálculo da assinatura faz-se tomando os parâmetros na ordem seguinte: **shopId, transmissionDate, transactionId, sequenceNb, ctxMode, autorisationNb,autorisationDate, comment**

Esta função retorna uma resposta do tipo Standard Response (cf 3.1). Os códigos de erros (errorCode) possíveis são:

errorCode	Descrição
0	Ação realizada com sucesso
1	Ação não autorizada
2	Transação não encontrada
3	Transação não colocada no Status correto
5	Assinatura errada
50	Parâmetro ‘shopId’ Inválido
51	Parâmetro ‘transmissionDate’ Inválido
52	Parâmetro ‘transactionId’ Inválido
53	Parâmetro ‘ctxMode’ Inválido
54	Parâmetro ‘comment’ Inválido
99	Erro desconhecido

NB: O código de erro extenso (extendedErrorCode) não está informado para esta função.

4.4. Modify

Esta função permite alterar o valor de uma transação (para baixo) ou alterar a data da captura desejada.
As transações que podem ser alteradas possuem um dos seguintes status:

- Para validar
- Para validar e autorizar
- Em espera
- Em espera d'auto
- Em espera da captura

Nome do campo	Tipo	Descrição	Obrigatório
shopId	String	Identificador da loja	X
transmissionDate	Date	Data da transação	X
transactionId	String	Identificador da transação	X
sequenceNb	Int	Número de sequência da transação	X
ctxMode	String	Contexto de ambiente da plataforma ("TEST", "PRODUCTION")	X
amount	Long	Valor do pedido incluindo casa decimal	X
devise	Int	Moeda (Código moeda ISO 4217, Real : 986)	X
remiseDate	Date	Data captura pedida	
comment	String	Comentário "livre"	X
wsSignature	String	Assinatura (cf §5)	X

O cálculo da assinatura faz-se tomando os parâmetros na ordem seguinte: **shopId, transmissionDate, transactionId, sequenceNb, ctxMode, amount, devise, remiseDate, comment**

Esta função retorna uma resposta do tipo Standard Response (cf 3.1). Os códigos de erros (errorCode) possíveis são :

errorCode	Descrição
0	Ação realizada com sucesso
1	Ação não autorizada
2	Transação não encontrada
3	Transação não colocada no Status correto
5	Assinatura errada
50	Parâmetro 'shopId' Inválido
51	Parâmetro 'transmissionDate' Inválido
52	Parâmetro 'transactionId' Inválido
53	Parâmetro 'ctxMode' Inválido
54	Parâmetro 'comment' Inválido
99	Erro desconhecido

NB: O código de erro extenso (extendedErrorCode) não está informado para esta função.

4.5. Refund

Esta função permite creditar o valor ao portador.

Esta função leva em consideração os parâmetros seguintes:

Nome do campor	Tipo	Descrição	Obrigatório
shopId	String	Identificador da loja	X
transmissionDate	Date	Data da transação	X
transactionId	String	Identificador da transação	X
sequenceNb	Int	Número de sequência da transação	X
ctxMode	String	Contexto de ambiente da plataforma("TEST", "PRODUCTION")	X
newTransactionId	String	Identificador da transação criada	X
amount	Long	Valor pedido incluindo casa decimal	X
devise	Int	Moeda (Código moeda ISO 4217, Real: 986)	X
remiseDate	Date	Data captura pedida	
validationMode	int	0 = Automático, 1 = Manual	
comment	String	Comentário "livre"	
wsSignature	String	Assinatura (cf §5)	X

O cálculo da assinatura faz-se tomando os parâmetros na ordem seguinte: **shopId, transmissionDate, transactionId, sequenceNb, ctxMode, newTransactionId, amount, devise, presentationDate, validationMode, comment**

Esta função retorna uma resposta do tipo TransactionInfo (cf 3.3). Os códigos de erros (errorCode) possíveis são:

errorCode	Descrição
0	Ação realizada com sucesso
1	Ação não autorizada
2	Transação não encontrada
3	Transação não colocada no Status correto
5	Assinatura errada
50	Parâmetro ‘shopId’ Inválido
51	Parâmetro ‘transmissionDate’ Inválido
52	Parâmetro ‘transactionId’ Inválido
53	Parâmetro ‘ctxMode’ Inválido
54	Parâmetro ‘comment’ Inválido
99	Erro desconhecido

NB: O código de erro extenso (extendedErrorCode) não está informado para esta função.

4.6. Duplicate

Esta função permite criar uma nova transação tendo exatamente as mesmas características da transação que serviu de base à duplicação.

Esta função leva em consideração os parâmetros seguintes:

Nome do Campo	Tipo	Descrição	Obrigatório
shopId	String	Identificador da loja	X
transmissionDate	Date	Data da transação	X
transactionId	String	Identificador datransação	X
sequenceNb	Int	Número de sequência da transação	X
ctxMode	String	Contexto de ambiente da plataforma("TEST", "PRODUCTION")	X
orderId	String	Referência do pedido	X
orderInfo	String	Descrição livre do pedido	
orderInfo2	String	Descrição livre do pedido	
orderInfo3	String	Descrição livre do pedido	
amount	Long	Valor incluindo casa decimal	X
devise	int	Moeda (Código moeda ISO 4217, Real : 986)	
newTransactionID	String	Identificador da transação criada	
presentationDate	Date	Data de captura pedida	
validationMode	int	0 = Automático, 1 = Manual	
comment	String	Comentário "livre"	
wsSignature	String	Assinatura (cf §5)	X

O cálculo da assinatura faz-se tomando os parâmetros na ordem seguinte: **shopId, transmissionDate, transactionId, sequenceNb, ctxMode, orderId, orderInfo, orderInfo2, orderInfo3, amount, devise, newTransactionId, presentationDate, validationMode, comment**

Esta função retorna uma resposta do tipo TransactionInfo (cf 3.3).

Os códigos de erros possíveis são:

errorCode	Descrição
0	Ação realizada com sucesso
1	Ação não autorizada
2	Transação não encontrada
3	Transação não colocada no status correto
4	TransactionId/Sequence/TransmissionDate já existente
5	Assinatura errada
10	Valor errado
11	Moeda errada
50	Parâmetro 'shopId' inválido
51	Parâmetro 'transmissionDate' inválido
52	Parâmetro 'transactionId' inválido
53	Parâmetro 'ctxMode' inválido
54	Parâmetro 'comment' inválido
57	Parâmetro 'presentationDate' inválido
58	Parâmetro 'newTransactionId' inválido
59	Parâmetro 'validationMode' inválido
60	Parâmetro 'orderId' inválido
61	Parâmetro 'orderInfo' inválido
62	Parâmetro 'orderInfo2' inválido

63	Parâmetro ‘orderInfo3’ inválido
99	Erro desconhecido

NB: O código de erro extenso (extendedErrorCode) não está informado para esta função.

4.7. Create

Esta função permite:

- Os pagamentos "manuais" (do tipo VAD – ERT 20), resultante dos diferentes canais;
- Os pagamentos "automáticos" para os quais a aquisição dos dados cartões é realizada pelo comerciante.

Esta função leva em consideração os parâmetros seguintes:

Nome do Campo	Tipo	Descrição	Obrigatório
createInfo	CreatePaiementInfo	Cf 3.4	X
wsSignature	String	Assinatura (cf § 5)	X

O cálculo da assinatura faz-se tomando os parâmetros na ordem seguinte: **shopId, transmissionDate, transactionId, paymentMethod, orderId, orderInfo, amount, devise, presentationDate, validationMode, cardNumber, cardNetwork, cardExpirationDate, cvv, contractNumbe, customerId, customerTitle, customerName, customerPhone, customerMail, customerAddress, customerZipCode, customerCity, customerCountry, customerLanguage, customerIP, customerSendEmail, ctxMode, comment**

Esta função devolve uma resposta do tipo TransactionInfoSig (cf 2.3).

Os códigos de erros (errorCode) possíveis são:

errorCode	Descrição
0	Ação realizada com sucesso
1	Ação não autorizada
4	TransactionId/Sequence/TransmissionDate já existente
5	Assinatura errada
10	Montante errado
11	Moeda errada
12	Tipo do cartão desconhecido
13	Data de vencimento do cartão incorreta
14	CVV obrigatório
15	Contrato desconhecido
16	Número do cartão errado (comprimento, luhn) …)
50	Parâmetro ‘shopId’ inválido
51	Parâmetro ‘transmissionDate’ inválido
52	Parâmetro ‘transactionId’ inválido
53	Parâmetro ‘ctxMode’ inválido
50	Parâmetro ‘shopId’ inválido
51	Parâmetro ‘transmissionDate’ inválido
52	Parâmetro ‘transactionId’ inválido
53	Parâmetro ‘ctxMode’ inválido
54	Parâmetro ‘comment’ inválido
57	Parâmetro ‘presentationDate’ inválido
58	Parâmetro ‘newTransactionId’ inválido
59	Parâmetro ‘validationMode’ inválido

60	Parâmetro ‘orderId’ inválido
61	Parâmetro ‘orderInfo’ inválido
62	Parâmetro ‘orderInfo2’ inválido
63	Parâmetro ‘orderInfo3’ inválido
64	Parâmetro ‘PaymentMethod’ inválido
65	Parâmetro ‘cardNetwork’ inválido
66	Parâmetro ‘contractNumber’ inválido
67	Parâmetro ‘customerId’ inválido
68	Parâmetro ‘customerTitle’ inválido
69	Parâmetro ‘customerName’ inválido
70	Parâmetro ‘customerPhone’ inválido
71	Parâmetro ‘customerMail’ inválido
72	Parâmetro ‘customerAddress’ inválido
73	Parâmetro ‘customerZipCode’ inválido
74	Parâmetro ‘customerCity’ inválido
75	Parâmetro ‘customerCountry’ inválido
76	Parâmetro ‘customerLanguage’ inválido
77	Parâmetro ‘customerIP’ inválido
78	Parâmetro ‘customerSendMail’ inválido
99	Erro desconhecido

NB: O código de erro extenso (extendedErrorCode) não está informado para esta função.

4.8. getInfo

Esta função permite interrogar uma transação para conhecer seus diferentes atributos.

Esta função leva em consideração os parâmetros seguintes:

Nome do Campo	Tipo	Descrição	Obrigatório
shopId	String	Identificador da loja	X
transmissionDate	Date	Data da transação	X
transactionId	String	Identificador da transação	X
sequenceNb	Int	Número de sequência da transação	X
ctxMode	String	Contexto de ambiente da plataforma("TEST", "PRODUCTION")	X
wsSignature	String	Assinatura (cf §5)	X

O cálculo da assinatura faz-se tomando os parâmetros na ordem seguinte: shopId, transmissionDate, transactionId, sequenceNb, ctxMode

Esta função retorna uma resposta do tipo TransactionInfo (cf 3.3).

Os códigos erros (errorCode) possíveis são:

errorCode	Descrição
0	Ação realizada com sucesso
1	Ação não autorizada
2	Transação não encontrada
5	Assinatura errada
50	Parâmetro 'shopId' inválido
51	Parâmetro 'transmissionDate' inválido
52	Parâmetro 'transactionId' inválido
53	Parâmetro 'ctxMode' inválido
99	Erro desconhecido

NB: O código de erro extenso (extendedErrorCode) não está informado para esta função

5. ASSINATURA

Um certificado é necessário para trocar informações com a plataforma de pagamento. Ele está disponível para todas as pessoas autorizadas à consulta dos certificados na sua ferramenta de gestão no PayZEN em Configuração >> Lojas >> Sua loja >> Aba Certificados.

Existem dois certificados diferentes: um para a plataforma de teste e uma para a plataforma de produção.

A assinatura deve ser gerada do seguinte modo:

- Criação de uma lista de caracteres representando a concatenação dos parâmetros, separados pelo caractere "+".
- Acrescente a esta lista um "certificado" numérico (de teste ou de produção conforme o contexto do ambiente).
- Codificação da lista resultante com o algoritmo SHA1.

A plataforma de pagamento efetuara obrigatoriamente a verificação da assinatura. É da responsabilidade do comerciante verificar por sua vez a assinatura transmitida no retorno.

A ordem dos campos deve ser respeitada.
Os campos do tipo data devem ser formatados da seguinte maneira: AAAAMMDD
Ou seja, para o 1° Fevereiro 2009: 20090201.

Os campos do tipo numérico não devem ter um 0 na esquerda do caractere mais significante. Os campos do tipo booleano tomam os seguintes valores:

- 1 para verdadeiro (true)
- 0 para falso (false)

Os campos do tipo String não informados estarão vázios.

Exemplo: Para uma chamada Cancel, se os parâmetros do pedido são os seguintes:

- shopId = 12345678
- transmissionDate = 7 Março 2012
- transactionId = 654321
- sequenceNb = 1
- ctxMode = TEST
- comment = não informado
-

Se o valor do certificado de teste é 1122334455667788, então a lista a ser utilizada para a Separação com a ajuda do algoritmo SHA1 é a seguinte: **12345678+20100307+654321+1+TEST++1122334455667788**

O que resulta depois da codificação SHA é: **88ff8fc1897ac4edf34c5e2327be5adde76e17d6**

6. EXEMPLO DE INTEGRAÇÃO

Este exemplo utiliza os serviços webservice client do Jboss.

6.1. Gerar stub a partir de wsdl

A fim de utilizar os serviços web, você precisa gerar o código que vai trocar informações com os serviços web:

wsconsume.bat -k -p com.lyra.vads.ws.stubs https://secure.payzen.com.br/vads-ws/v2?wsdl

6.2. Exemplo de código para gerar a assinatura

```
import java.security.MessageDigest;
import java.text.SimpleDateFormat;
import java.util.Date;

public class Signature {

   static public final String SEPARATOR = "+";
   static final String key = "1122334455667788";

   public static String createSignature(Object... params) {

      StringBuilder builder = new StringBuilder();

      for (int i = 0; i < params.length; i++) {
         if (i != 0) {
            builder.append(SEPARATOR);
         }
         if (params[i] != null) {
            if (params[i] instanceof Date) {
               Date date = (Date) params[i];
               SimpleDateFormat myDateFormat = new SimpleDateFormat("yyyyMMdd");
               builder.append(myDateFormat.format(date));
            } else if (params[i] instanceof Boolean) {
               if ((Boolean) params[i])
                  builder.append(1);
               else
                  builder.append(0);
            } else {
               builder.append(params[i]);
            }
         }
      }
      String toSign = builder.toString() + SEPARATOR + key;
      return encode(toSign);
   }
   public static String encode(String src) {
      try {
         MessageDigest md;
         md = MessageDigest.getInstance("SHA-1");
```

```
            byte bytes[] = src.getBytes("iso-8859-1");

            md.update(bytes, 0, bytes.length);
            byte[] sha1hash = md.digest();

            return convertToHex(sha1hash);
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    }
    private static String convertToHex(byte[] sha1hash) {
        StringBuilder builder = new StringBuilder();
        for (int i = 0; i < sha1hash.length; i++) {
            byte c = sha1hash[i];

            addHex(builder, (c >> 4) & 0xf);
            addHex(builder, c & 0xf);
    }
    return builder.toString();
}
private static void addHex(StringBuilder builder, int c) {
        if (c < 10)
        builder.append((char) (c + '0'));
        else
        builder.append((char) (c + 'a' - 10));
    }
}
```

6.3. Exemplo de código para canelar um pagamento

```
public static void main(String[] args) throws Exception {
    URL wsdlURL = new URL(
        "https://secure.payzen.com.br/vads-ws/v3?wsdl");

    QName qname = new QName("http://v3.ws.vads.lyra.com/", "StandardWS");
    Service service = Service.create(wsdlURL, qname);
    StandardWs port = service.getPort(StandardWs.class);

    GregorianCalendar gCalendar = new GregorianCalendar();
    gCalendar.setTime(new Date());
    XMLGregorianCalendar xmlCalendar = DatatypeFactory.newInstance()
        .newXMLGregorianCalendar(gCalendar);
    XMLGregorianCalendar expDate = DatatypeFactory.newInstance()
        .newXMLGregorianCalendar(2010, 12, 31, 0, 0, 0, 0, 0);

    CreatePaiementInfo myPayment = new CreatePaiementInfo();

    // Código da loja
    myPayment.setShopId("XXXXXXXX");
    // Data da transmissão
    myPayment.setTransmissionDate(xmlCalendar);
    // Número de transação
    myPayment.setTransactionId("123456");
    // VPC
```

```
    myPayment.setPaymentMethod("EC");

    // Código do pedido
    myPayment.setOrderId("maCommande");
    // Informação livre do pedido => Facultatif !
    myPayment.setOrderInfo("info commande");
    // Valor incluindo os centavos
    myPayment.setAmount(3000);
    // Moeda => 986 Real
    myPayment.setDevise(978);
    // Data da captura
    myPayment.setPresentationDate(xmlCalendar);
    // Validação automática
    myPayment.setValidationMode(0);

    // Número do cartão de crédito
    myPayment.setCardNumber("4970100000000000");
    // Rede adquirente
    myPayment.setCardNetwork("CB");
    // Data de validade do cartão
    myPayment.setCardExpirationDate(expDate);
    // CVV
    myPayment.setCvv("000");

    // Contexto do ambiente (TEST ou PRODUCTION)
    myPayment.setCtxMode("TEST");
    // Um comentário livre
    myPayment.setComment("Creation Par WebService");


    String signature = Signature.createSignature(myPayment.getShopId(),
       myPayment.getTransmissionDate(),myPayment.getTransactionId(),
       myPayment.getPaymentMethod(), myPayment.getOrderId(),
       myPayment.getOrderInfo(), myPayment.getOrderInfo2(),
       myPayment.getOrderInfo3(), myPayment.getAmount(),
       myPayment.getDevise(), myPayment.getPresentationDate(),
       myPayment.getValidationMode(), myPayment.getCardNumber(),
       myPayment.getCardNetwork(), myPayment.getCardExpirationDate(),
       myPayment.getCvv(), myPayment.getContractNumber(), null,
       myPayment.getSubPaymentType(), myPayment.getSubReference(),
       myPayment.getSubPaymentNumber(), myPayment.getCustomerId(),
       myPayment.getCustomerTitle(), myPayment.getCustomerName(),
       myPayment.getCustomerPhone(), myPayment.getCustomerMail(),
       myPayment.getCustomerAddress(), myPayment.getCustomerZipCode(),
       myPayment.getCustomerCity(), myPayment.getCustomerCountry(),
       myPayment.getCustomerLanguage(), myPayment.getCustomerIP(),
       myPayment.isCustomerSendEmail(), myPayment.getCtxMode(),
       myPayment.getComment());

    TransactionInfo respTransactionInfo = port.create(myPayment,signature);

    System.out.println("WS Result : " + respTransactionInfo.getErrorCode()+ " / " +
respTransactionInfo.getExtendedErrorCode() + " / " + respTransactionInfo.getTransactionStatus());

}
```

7. CONTATOS

Qualquer dúvida sobre as funcionalidades e manuseio operacional, entre em contato.

Reinaldo Santos
Tel : 11 3336-9200
Fax : 11 8716-0037
Email : reinaldo.santos@lyra-network.com.br