

# **Correios *Web Service* (CWS)**

## **Padrões Técnicos de Comunicação do *Serviço Correios Log – Comércio Eletrônico e-fulfillment***

**Versão 1.9**

## Tabela de Histórico da Evolução do Documento

Data	Evento	Versão	Gerência/AC Seção/DR	Responsável
14/10/2016	Criação	1.0	DECOR/GFCS	Vanessa Pedroso
25/10/2016	Atualização dos recursos disponibilizados na 1ª fase	1.1	DECOR/GFCS	Vanessa Pedroso
14/11/2016	Atualização das mensagens de retorno e das regras de validação	1.2	DECOR/GFCS	Vanessa Pedroso
08/12/2016	Inclusão de novo recurso e atualização das mensagens de retorno e regras de validação	1.3	DECOR/GFCS	Vanessa Pedroso
15/03/2017	Revisão do gestor funcional	1.4	VILOG/DEMKT/GCME	Helton Lucio da Silva Soares
03/04/2017	Inclusão do método “consultaPedido”	1.5	DEGOR/GRLO	Karine de Mendonça Silva Monteiro
24/04/2017	Inclusão do recurso “Emitir NFe”	1.6	DECOR/GFCS	Gilson Oliveira
28/07/2017	Inclusão dos códigos do produtos cadastrados no ambiente de homologação	1.7	DEREL/GROP	Karine de Mendonça Silva Monteiro
15/08/2017	Inclusão do componente “enviar NF-e de venda” e melhorias de leiaute	1.8	DEREL/GROP	Karine de Mendonça Silva Monteiro
08/09/2017	Atualização do componente “inserir pedido”	1.9	DEREL/GROP	Karine de Mendonça Silva Monteiro

## Tabela de Revisão do Documento

Papel	Nome do Profissional Revisor	Data Revisão	Versão Revisada	Assinatura

## Sumário

1.	Introdução .....	5
2.	Correios Web Services .....	5
3.	Autenticação e Autorização de Acesso .....	6
4.	Leiautes .....	7
5.	Códigos de status .....	8
6.	Serviços Correios Log - e-fulfillment .....	12
6.1	Consulta o estoque de um produto .....	12
6.1.1	Leiaute dos parâmetros de entrada .....	12
6.1.2	Leiaute do retorno .....	13
6.1.3	Mensagens do retorno .....	13
6.2	Criar um novo pedido .....	13
6.2.1	Leiaute dos parâmetros de entrada: .....	15
6.2.2	Leiaute do retorno: .....	16
6.2.3	Regras de Negócios aplicadas: .....	17
6.3	Consultar pedido .....	17
6.3.1	Leiaute dos parâmetros de entrada: .....	18
6.3.2	Leiaute do retorno: .....	18
6.4	Enviar o XML da Nota Fiscal Eletrônica relativa ao Pedido enviado .....	20
6.4.1	Leiaute dos parâmetros de entrada: .....	20
6.4.2	Leiaute do retorno: .....	20

6.4.3 Mensagens do retorno .....	21
7. Exemplo de utilização de um webservice REST .....	21
7.1 Apresentação .....	21
7.2 Cliente Java.....	21
7.3 Conversão.....	27
8. Dúvidas .....	29

## 1. Introdução

Os Correios *Web Service* (CWS) tem o objetivo de fornecer uma plataforma de serviços, baseados na tecnologia de *Web Services*, que disponibiliza suas principais informações aos clientes do Comércio Eletrônico Brasileiro. Estes serviços permitem que as soluções customizadas de TI dos clientes sejam facilmente integradas aos recursos disponíveis pelos Correios com o intuito de agilizar suas operações, simplificar seus processos e com qualidade no atendimento.

## 2. Correios Web Services

A arquitetura de software dos serviços fornecidos pelos Correios *Web Service* (CWS) seguem os padrões e os protocolos de comunicação descritos acima, conforme mostra a Figura 1 de exemplo.

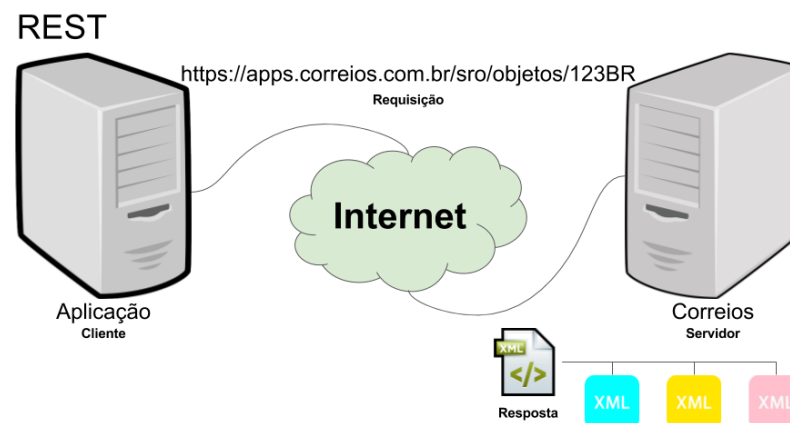


Figura 1 – Arquitetura do CWS

O serviço que será detalhado neste documento utiliza a arquitetura REST (*REpresentational State Transfer*) por meio de requisições HTTP e transferência de dados no formato XML ou JSON.

### **3. Autenticação e Autorização de Acesso**

Os componentes do serviço Correios Log (e-fulfillment) necessitam apenas de autenticação para acessá-los. Para isso, os clientes precisam criar um idCorreios nos seguintes endereços, a depender do ambiente a ser utilizado:

URL de homologação: <https://apphom.correios.com.br/idCorreios>

URL de produção: <http://apps.correios.com.br/idCorreios>

O idCorreios é um mecanismo de autenticação e autorização única de usuários para acesso ao Portal Correios e seus serviços. O serviço é disponibilizado gratuitamente pelos Correios para os clientes e usuários do seu site internet e permite acesso seguro e padronizado, eliminando a necessidade de se informar um *login* e uma senha para cada serviço utilizado.

No cadastro do idCorreios para Pessoa Jurídica deverá informar o CNPJ do contrato que está sendo utilizado nos Correios. Após a finalização do cadastro, deve ser acessado o menu "Componentes com permissão de acesso recebida" e criar uma senha de acesso aos componentes/serviços.

Desta forma, o acesso para autenticação e autorização dos serviços Correios Log (e-fulfillment) é constituído de um identificador alfanumérico e uma senha de acesso aos componentes/serviços, definidos pelo usuário.

Para realização de testes no ambiente de homologação, deve ser usado os seguintes dados, conforme abaixo:

Login: correioslogtest

Senha: log123789

Cartão número: 0067599060

Para ambiente de produção, o cliente necessita de um número de cartão de postagem a ser disponibilizado pelo representante comercial dos Correios. Neste cartão de postagem (Cartão Correios Fácil), deverão estar vinculados os códigos do serviço Correios Log – Comércio Eletrônico (e-fulfillment).

Para as plataformas de ERP e empresas de tecnologia que desejam realizar a integração com os Correios e que não possuem um contrato comercial celebrado, essas devem utilizar o número do cartão acima para o ambiente de homologação. Para o ambiente de produção, elas devem utilizar um número de um cartão de postagem pertencente a uma empresa que de fato, utilizará o serviço. Esse número de cartão é informado pelas empresas que possuem contrato comercial com os Correios. Para contatos comerciais a respeito da geração ou consulta desse Cartão de Postagem definitivo (Cartão Correios Fácil), as empresas podem enviar um e-mail para:

Estados: SP, RJ, PR, SC, RS: [logisticaspm@correios.com.br](mailto:logisticaspm@correios.com.br)

Demais estados: [logisticabsb@correios.com.br](mailto:logisticabsb@correios.com.br)

Para o ambiente de produção, o cliente portador do contrato comercial dos Correios que utilizará o serviço, deverá se cadastrar no idCorreios e informar à sua plataforma de ERP escolhida, o login e senha gerados no idCorreios.

## 4. Leiautes

Sobre os leiautes, segue a explicação de cada coluna:

**Item:** código do campo;

**Campo:** nome do campo no padrão xml;

**Descrição:** descrição do campo;

**Tipo:** “C” para campos alfanuméricos;

“N” para campos numéricos;

“D” para campos do tipo Data;

“B” para campos do tipo booleano (valor retornado *true* ou *false*);

“E” para campos enumerados (tabelados);

“G” para grupo (o grupo pode conter 1 ou mais campos);

“X” para campos que representam conteúdo do XML.

**Obs.:** valores numéricos que possuem zero à esquerda obrigatoriamente devem ser do tipo “C”.

**Ocorrência:** é o número mínimo e máximo de ocorrência do campo para um determinado grupo. Exemplos:

- 0-N: é opcional a existência do campo e, caso exista, no máximo N vezes;
- 1-N: é obrigatório ter o campo pelo menos uma vez e no máximo N vezes.

**Tamanho:** Tamanho da parte inteira no campo;

**Decimal:** Tamanho da parte decimal;

**Observações:** Mascara e/ou outros comentários.

## 5. Códigos de status

Toda requisição que é enviada para o servidor retorna um código de status. Esses códigos são divididos em cinco famílias: 1xx, 2xx, 3xx, 4xx e 5xx, sendo:

1xx – Informacionais;

2xx – Códigos de sucesso;

3xx – Códigos de redirecionamento;

4xx – Erros causados pelo cliente;

5xx – Erros originados no servidor.



Alguns dos códigos mais importantes e mais utilizados são:

**2xx****200 – OK**

Indica que a operação indicada teve sucesso.

**201 – Created**

Indica que o recurso desejado foi criado com sucesso. Deve retornar um cabeçalho Location, que deve conter a URL onde o recurso recém-criado está disponível.

**202 – Accepted**

Indica que a solicitação foi recebida e será processada em outro momento. É tipicamente utilizada em requisições assíncronas, que não serão processadas em tempo real. Por esse motivo, pode retornar um cabeçalho Location, que trará uma URL onde o cliente pode consultar se o recurso já está disponível ou não.

**204 – No Content**

Usualmente enviado em resposta a uma requisição PUT, POST ou DELETE, onde o servidor pode recusar-se a enviar conteúdo.

**3xx****301 – Moved Permanently**

Significa que o recurso solicitado foi realocado permanentemente. Uma resposta com o código 301 deve conter um cabeçalho Location com a URL completa (ou seja, com descrição de protocolo e servidor) de onde o recurso está atualmente.

**303 – See Other**

É utilizado quando a requisição foi processada, mas o servidor não deseja enviar o resultado do processamento. Ao invés disso, o servidor envia a resposta com este código de status e o cabeçalho Location, informando onde a resposta do processamento está.

**304 – NotModified**

É utilizado principalmente em requisições GET condicionais – quando o cliente deseja ver a resposta apenas se ela tiver sido alterada em relação a uma requisição anterior.

#### 307 – Temporary Redirect

Similar ao 301, mas indica que o redirecionamento é temporário, não permanente.

#### **4xx**

#### 400 – Bad Request

É uma resposta genérica para qualquer tipo de erro de processamento cuja responsabilidade é do cliente do serviço.

#### 401 – Unauthorized

Utilizado quando o cliente está tentando realizar uma operação sem ter fornecido dados de autenticação (ou a autenticação fornecida for inválida).

#### 403 – Forbidden

Utilizado quando o cliente está tentando realizar uma operação sem ter a devida autorização.

#### 404 – Not Found

Utilizado quando o recurso solicitado não existe.

#### 405 – Method Not Allowed

Utilizado quando o método HTTP utilizado não é suportado pela URL. Deve incluir um cabeçalho Allow na resposta, contendo a listagem dos métodos suportados (separados por “,”).

#### 409 – Conflict

Utilizado quando há conflitos entre dois recursos. Comumente utilizado em resposta a criações de conteúdos que tenham restrições de dados únicos – por exemplo, criação de um usuário no sistema utilizando um login já existente. Se for causado pela existência de outro recurso (como no caso citado), a resposta deve conter um cabeçalho Location, explicitando a localização do recurso que é a fonte do conflito.

#### 410 – Gone

Semelhante ao 404, mas indica que um recurso já existiu neste local.

#### 415 – UnsupportedMedia Type

Utilizado em resposta a clientes que solicitam um tipo de dados que não é suportado – por exemplo, solicitar JSON quando o único formato de dados suportado é XML.

### **5xx**

#### 500 – Internal Server Error

É uma resposta de erro genérica, utilizada quando nenhuma outra se aplica.

#### 503 – Service Unavailable

Indica que o servidor está atendendo requisições, mas o serviço em questão não está funcionando corretamente. Pode incluir um cabeçalho Retry-After, dizendo ao cliente quando ele deveria tentar submeter a requisição novamente.

## 6. Serviços Correios Log - e-fulfillment

### 6.1 Consulta o estoque de um produto

URL de homologação: <https://apphom.correios.com.br/efulfillment/v1/produtos/{codigo}/estoque> (GET)

URL de produção: <https://cws.correios.com.br/efulfillment/v1/produtos/{codigo}/estoque> (GET)

Envio:

HEADER: numeroCartaoPostagem (string)

#### 6.1.1 Leiaute dos parâmetros de entrada

Campo	Descrição	Tipo	Obrigatório	Tamanho	Decimal	Observações
numeroCartaoPostagem	Número do cartão de postagem	C	S	10		O cartão de postagem deve pertencer ao usuário cadastrado no idCorreios. O número do cartão de postagem é fornecido pelo representante Comercial dos Correios, após a celebração do contrato Múltiplo ou inclusão do Anexo do serviço à um contrato já existente.
codigo	Código de produto	C	S	60		O produto deve pertencer ao contrato do cliente cadastrado no idCorreios. No ambiente de homologação podem ser utilizados os seguintes códigos de produtos: 27849C5921L 27849C5920L 27849C5921M 27367C0525L
armazem	Código do armazém dos Correios	C	S	12		Cada armazém dos Correios possui um código. No arquivo, deve constar o código do armazém escolhido pelo cliente para sua operação. Os códigos são:

						- Brasília/DF: 00430112; - Cajamar/SP: 00426980; - Contagem/MG: 00425002; - Rio de Janeiro/RJ: 00430158; - Curitiba/PR: 00425009; - Recife/PE: 00425007.
--	--	--	--	--	--	---

### 6.1.2 Leiaute do retorno

Campo	Descrição	Tipo	Obrigatório	Tamanho	Decimal	Observações
total	Quantidade bruta de produtos no estoque do armazém (WMS)	C	S	10		
reservado	Quantidade de produtos reservados no sistema.	C	S	10		
disponível	Quantidade de produtos disponíveis para venda.	C	S	10		Disponível = total – reservado

### 6.1.3 Mensagens do retorno

Código	Mensagem
400	Cartão de postagem inválido
404	Cartão de postagem não encontrado
	Produto não encontrado
	Estoque do produto não encontrado
	Código do armazém não encontrado

## 6.2 Criar um novo pedido

URL de homologação: <https://apphom.correios.com.br/efulfillment/v1/pedidos> (POST)

URL de produção: <https://cws.correios.com.br/efulfillment/v1/pedidos> (POST)

Envio:

HEADER: numeroCartaoPostagem (string)

BODY:

```
{
  "codigoArmazem": "string",
  "numero": "string",
  "dataSolicitacao": "string",
  "valordeclarado": "string",
  "cartaoPostagem": "string",
  "codigoservico": "string",
  "numeroSerie": "string",
  "servicosAdicionais": ["string"],
  "cnpjTransportadora": "string",
  "destinatario": {
    "nome": "string",
    "logradouro": "string",
    "numeroEndereco": "string",
    "complemento": "string",
    "bairro": "string",
    "cep": "string",
    "cidade": "string",
    "uf": "string",
    "ddd": "string",
    "telefone": "string",
```

```
"email": "string",  
"cnpj": "string",  
"cpf": "string",  
,  
"itensPedido": [  
  {  
    "codigo": "string",  
    "quantidade": "string"  
  }  
,  
]  
}
```

### 6.2.1 Leiaute dos parâmetros de entrada:

Campo	Descrição	Tipo	Obrigatório	Tamanho	Decimal	Observações
<b>HEADER</b>						
numeroCartaoPostagem	Código do cartão de postagem	C	S	10		Conforme orientações fornecidas na página 5 e 6, item 3.
<b>BODY</b>						
pedidos	Dados do pedido					
codigoArmazem	Código do armazém dos Correios	C	S	12		Cada armazém dos Correios possui um código. No arquivo, deve constar o código do armazém escolhido pelo cliente para sua operação. Os códigos são: - Brasília/DF: 00430112; - Cajamar/SP: 00426980; - Contagem/MG: 00425002; - Rio de Janeiro/RJ: 00430158; - Curitiba/PR: 00425009; - Recife/PE: 00425007.
numero	Número do pedido do cliente	C	S	12		
dataSolicitacao	Data de solicitação do pedido	C	S			DD/MM/AAAA HH:MM:SS

valordeclarado	Valor declarado para o serviço adicional	C	N	13		Obrigatório para o serviço adicional 019
codigoServico	Código do serviço	C	S	15		PAC – CATEGORIA LOGÍSTICA: 39217 – e-fulfillment entrega econômica:  SEDEX – CATEGORIA LOGÍSTICA: 39012 – e-fulfillment entrega expressa
numeroSerie	Número de Série do pedido do cliente	C	N			Número de série do pedido do cliente.
servicosAdicionais	Códigos dos serviços adicionais	C	N	3		Máximo 5 serviços adicionais
cnpjTransportadora	CNPJ da Transportadora	C	N			Quando vier em branco, será assumido, o CNPJ dos Correios.
destinatario	Destinatário					
nome	Nome	C	S	60		
logradouro	Logradouro	C	S	72		
numeroEndereco	Número do endereço	C	N	6		
complemento	Complemento do logradouro	C	N	60		
bairro	Bairro	C	S	72		
cep	CEP	C	S	8		
cidade	Cidade	C	S	40		
uf	UF	C	S	2		
ddd	DDD do telefone	C	N	3		
telefone	Telefone	C	N	10		
email	E-mail	C	N	200		
cpf	CPF do destinatário	C	N	11		Obrigatório para pessoa física
cnpj	CNPJ do destinatário	C	N	11		Obrigatório para pessoa jurídica
itensPedido	Lista de produtos					
codigo	Código do produto	C	S	60		No ambiente de homologação podem ser utilizados os seguintes códigos de produtos: 27849C5921L 27849C5920L 27849C5921M 27367C0525L
quantidade	Quantidade do produto	C	S	10		

## 6.2.2 Leiaute do retorno:

Caso Pedido criado com sucesso, retorna o código 201 e a URL do recurso criado:



<http://cws.correios.com.br/efulfillment/v1/pedidos/<numeroPedidoCorreios><número do pedido dos Correios>>

Guarde o número do pedido dos Correios, pois ele será necessário para resgatar os dados do pedido, incluir o XML da Nota Fiscal de Venda, etc.

Exemplo: <http://cws.correios.com.br/efulfillment/v1/pedidos/<numeroPedidoCorreios>249422>

Para demais retornos, verificar códigos de status (vide item 5 deste documento).

### 6.2.3 Regras de Negócios aplicadas:

- 1) Cartão de Postagem deve ser válido;
- 2) O código do serviço deve ser válido e estar atrelado ao cartão de postagem informado;
- 3) CEP deve ser válido e conferir com a cidade e/ou UF informada;
- 4) E-mail deve ser válido;
- 5) Produto deve existir e estar vinculado ao depositante;
- 6) Estoque do produto deve ser suficiente;
- 7) O código do armazém deve ser válido;
- 8) Código do serviço deve ser válido e estar cadastrado no cartão de postagem do cliente;
- 9) CPF e CNPJ deve ser válidos quando informado.

## 6.3 Consultar pedido

URL de homologação: <https://apphom.correios.com.br/efulfillment/v1/pedidos/<número do pedido dos Correios>> (GET)

URL de produção: <https://cws.correios.com.br/efulfillment/v1/pedidos/<número do pedido dos Correios>> (GET)

**IMPORTANTE:** O número do pedido dos Correios será obtido no retorno do método para criar o Pedido (vide item 6.2).

Envio:

HEADER: numeroCartaoPostagem (string)

### 6.3.1 Leiaute dos parâmetros de entrada:

Campo	Descrição	Tipo	Obrigatório	Tamanho	Decimal	Observações
<b>HEADER</b>						
numeroCartaoPostagem	Código do cartão de postagem	C	S	10		Conforme orientações fornecidas na página 5 e 6, item 3.
<b>Enviado na URL</b>						
numero	Número do pedido - retornado pelos Correios	C	S	08		Número do pedido retornado pelos Correios no componente "inserir pedido". Esse parâmetro deve ser passado na URL da requisição.

### 6.3.2 Leiaute do retorno:

Campo	Descrição	Tipo	Tamanho	Decimal	Observações
codigoArmazem	Código do armazém dos Correios	C	12		Os códigos são: - Brasília/DF: 00430112; - Cajamar/SP: 00426980; - Contagem/MG: 00425002; - Rio de Janeiro/RJ: 00430158; - Curitiba/PR: 00425009; - Recife/PE: 00425007.
numero	Número do pedido do cliente	C	08		
dataSolicitacao	Data de solicitação do pedido	C			DD/MM/AAAA
valorDeclarado	Valor declarado para o serviço adicional	C	13		
cartaoPostagem	Código do cartão de postagem	C	10		
codigoServico	Código do serviço	C	15		
numeroSerie	Número de série do pedido do cliente	C			Número de série do pedido do cliente.

servicosAdicionais	Códigos dos serviços adicionais	C	3		Será retornado quando houver.
cnpjTransportadora	Número do CNPJ da transportadora	C			Número de série da transportadora.
destinatario	Destinatário				
nome	Nome	C	60		
logradouro	Logradouro	C	72		
numeroEndereco	Número do endereço	C	6		
complemento	Complemento do logradouro	C	60		
bairro	Bairro	C	72		
cep	CEP	C	8		
cidade	Cidade	C	40		
uf	UF	C	2		
ddd	DDD do telefone	C	3		
telefone	Telefone	C	10		
email	E-mail	C	200		
cpf	CPF do destinatário	C	11		
cnpj	CNPJ do destinatário	C	11		
itensPedido	Lista de produtos				
codigo	Código do produto	C	60		
Quantidade	Quantidade do produto	C	10		
Situação	Situação do pedido	G			
codigoSituacao		C	2		Situações possíveis:  S - Solicitado V - Validado A - Aprovado EA - Em Atendimento EE - Em Expedição E - Expedido EI - Entregue - IDA X - Concluído C - Cancelado
Descricao		C			
Rastreo	Lista dos códigos dos objetos do pedido				
codigoObjeto	Código do objeto do pedido para rastreamento	C	13		
dataPostagem	Data da postagem do objeto	C			DD/MM/AAAA

## 6.4 Enviar o XML da Nota Fiscal Eletrônica relativa ao Pedido enviado

URL de homologação: <https://apphom.correios.com.br/efulfillment/v1/xmldanfedpedido/xml> (POST)

URL de produção: <https://cws.correios.com.br/efulfillment/v1/xmldanfedpedido/xml> (POST)

### 6.4.1 Leiaute dos parâmetros de entrada:

Campo	Descrição	Tipo	Obrigatório	Tamanho	Decimal	Observações
<b>HEADER</b>						
numeroCartaoPostagem	Código do cartão de postagem	C	S	10		Conforme orientações fornecidas na página 5 e 6, item 3.
armazem	Código do armazém dos Correios	C	S	12		Cada armazém dos Correios possui um código. No arquivo, deve constar o código do armazém escolhido pelo cliente para sua operação. Os códigos são: - Brasília/DF: 00430112; - Cajamar/SP: 00426980; - Contagem/MG: 00425002; - Rio de Janeiro/RJ: 00430158; - Curitiba/PR: 00425009; - Recife/PE: 00425007.
<b>Fomulário</b>						
XML	XML com informações da Nota Fiscal	C	S			XML padrão da Receita Federal.  <b>Constar na tag XPED o número do Pedido dos Correios referente à Nota Fiscal. A tag XPED deve ser inserida na TAG &lt;COMPRA&gt;.</b>

### 6.4.2 Leiaute do retorno:

Caso o XML da Nota Fiscal seja recebido com sucesso, retorna o código 200.

Para demais retornos, verificar códigos de status (vide item 5 deste documento).

### 6.4.3 Mensagens do retorno

Código	Mensagem
400	Cartão de postagem inválido
404	Cartão de postagem não encontrado
	Produto não encontrado
	Estoque do produto não encontrado
	Código do armazém não encontrado

## 7. Exemplo de utilização de um webservice REST

### 7.1 Apresentação

Este tutorial tem por objetivo apresentar um exemplo de uso dos serviços REST (REpresentational State Transfer) disponibilizados pelos Correios.


### 7.2 Cliente Java

Usaremos a linguagem de programação Java para criar um cliente capaz de manipular os dados do serviço (exemplo) de geografia das unidades de distribuição. Para isso, escreva uma classe com apenas um método main.

```
1 package br.com.correios.exemplo;
2
3 public class Cliente {
4
5     public static void main(String[] args) {
6
7     }
8
9 }
10
```

Imagem 1: Cliente Java


No interior do método *main*, escreva uma variável para armazenar a seguinte URL, utilizada como exemplo: <http://app.correiosnet.int/geo/unidades/00057878>.



```
1 package br.com.correios.exemplo;
2
3 public class Cliente {
4
5     public static void main(String[] args) {
6         String url = "http://app.correiosnet.int/geo/unidades/00057878";
7     }
8 }
9
10 }
11
```

Imagem 2: URL para acesso aos dados de uma unidade de distribuição

Há várias formas de executar uma requisição HTTP via código Java. Bibliotecas gratuitas e pagas estão disponíveis no mercado com essa finalidade, mas também é possível usar a própria API nativa do Java. Neste exemplo, usaremos a biblioteca Apache HttpClient (<https://hc.apache.org/>) por ser simples e gratuita.



```
1 package br.com.correios.exemplo;
2
3 import org.apache.http.client.methods.HttpGet;
4
5 public class Cliente {
6
7     public static void main(String[] args) {
8         String url = "http://app.correiosnet.int/geo/unidades/00057878";
9
10        HttpGet request = new HttpGet(url);
11    }
12 }
13
14 }
15
```

Imagem 3: Apache HttpClient

É preciso digitar o usuário e senha da sua aplicação. No entanto, não é possível digitá-los diretamente no cliente que estamos implementando. Antes, é necessário convertê-los em uma sequência única de caracteres, chamada de Token.

Para exemplificar, acesse o site BASE64 (<https://www.base64decode.org/>), selecione a aba “Encode” e verifique a existência de duas caixas de texto. Considere o usuário fictício “1234” e a senha fictícia “teste”. Na primeira caixa de texto, digite o usuário e senha da seguinte maneira:

**1234:teste**

Em seguida, clique em “> ENCODE <” e copie o Token gerado na segunda caixa de texto.

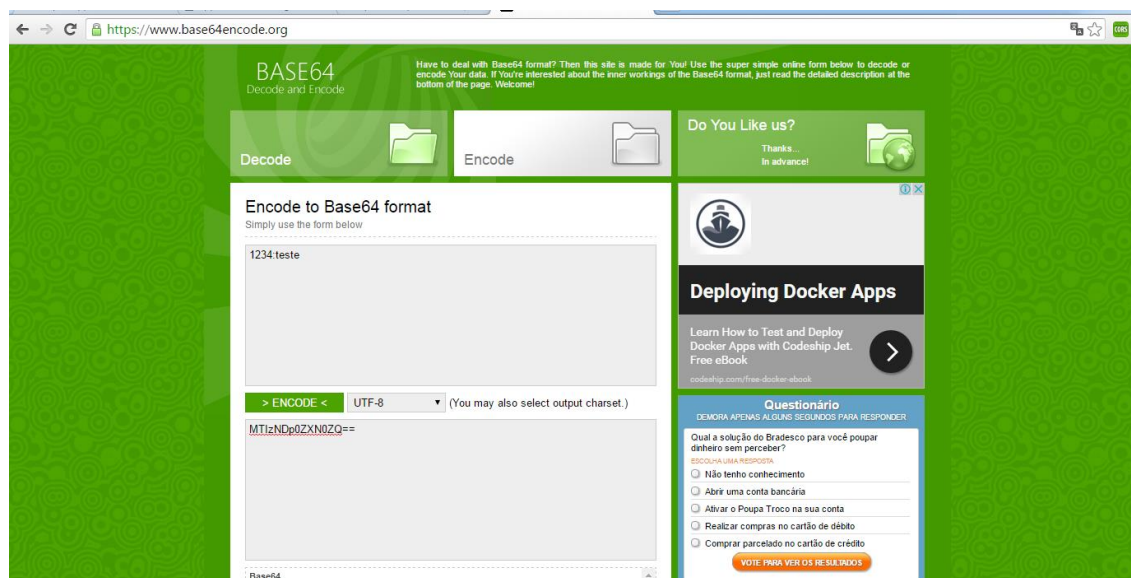
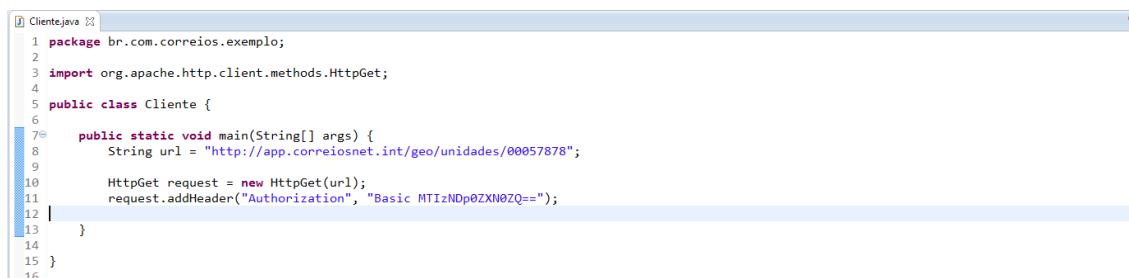


Imagem 4: BASE64

O Token gerado será adicionado ao cabeçalho da requisição, garantindo a autenticação para acesso aos dados do serviço.



```
1 package br.com.correios.exemplo;
2
3 import org.apache.http.client.methods.HttpGet;
4
5 public class Cliente {
6
7     public static void main(String[] args) {
8         String url = "http://app.correiosnet.int/geo/unidades/00057878";
9
10        HttpGet request = new HttpGet(url);
11        request.addHeader("Authorization", "Basic MTIzNDp0ZXN0ZQ==");
12    }
13 }
14
15 }
16
```

Imagem 5: Cabeçalho da requisição

Finalmente, podemos executar a requisição HTTP usando a biblioteca Apache HttpClient.



```
1 package br.com.correios.exemplo;
2
3 import java.io.IOException;
4
5 public class Cliente {
6
7     public static void main(String[] args) throws ClientProtocolException, IOException {
8         String url = "http://app.correiosnet.int/geo/unidades/00057878";
9
10        HttpGet request = new HttpGet(url);
11        request.addHeader("Authorization", "Basic MTIzNDp0ZXN0ZQ==");
12
13        HttpClient client = new DefaultHttpClient();
14        client.execute(request);
15    }
16 }
17
18 }
19
20 }
21
22 }
23
24 }
```

Imagem 6: Requisição HTTP

Dentre as funcionalidades oferecidas pela biblioteca, podemos visualizar o status da requisição. Lembrando que o status 200 indica uma requisição bem-sucedida.



```
1 package br.com.correios.exemplo;
2
3 import java.io.IOException;
4
5 public class Cliente {
6
7     public static void main(String[] args) throws ClientProtocolException, IOException {
8         String url = "http://app.correiosnet.int/geo/unidades/00057878";
9
10        HttpGet request = new HttpGet(url);
11        request.addHeader("Authorization", "Basic MTIzNDp0ZXN0ZQ==");
12
13        HttpClient client = new DefaultHttpClient();
14        HttpResponse response = client.execute(request);
15
16        System.out.println("Status Code: " + response.getStatusLine().getStatusCode());
17    }
18 }
```

Imagem 7: Status da requisição

Outra funcionalidade importante é o acesso à resposta da requisição, ou seja, os dados do serviço.

```
1 package br.com.correios.exemplo;
2
3 import java.io.IOException;
4
5 public class Cliente {
6
7     public static void main(String[] args) throws ClientProtocolException, IOException {
8         String url = "http://app.correiosnet.int/geo/unidades/00057878";
9
10        HttpGet request = new HttpGet(url);
11        request.addHeader("Authorization", "Basic MTIzNDp0ZXN0ZQ==");
12
13        HttpClient client = new DefaultHttpClient();
14        HttpResponse response = client.execute(request);
15
16        int statusCode = response.getStatusLine().getStatusCode();
17
18        if (statusCode == 200) {
19            InputStream content = response.getEntity().getContent();
20
21            Reader reader = new InputStreamReader(content);
22            System.out.println("XML: " + CharStreams.toString(reader));
23        }
24    }
25 }
```

Imagem 8: Resposta da requisição

Veja a resposta da requisição no console da sua IDE.

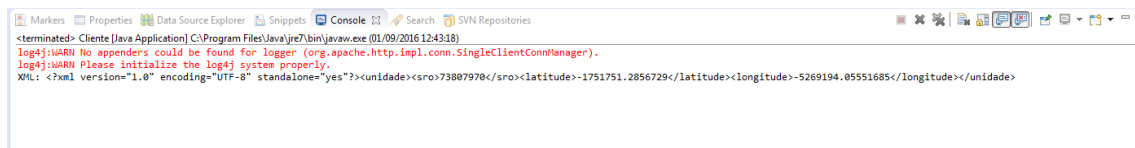
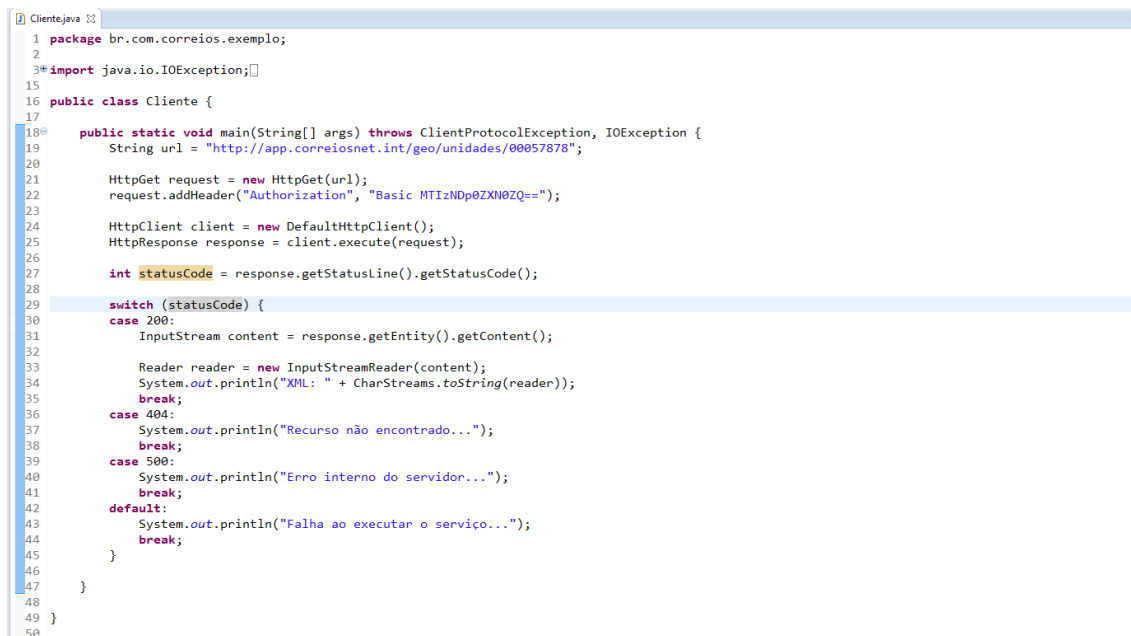


Imagem 9: Dados do serviço

Tratar os possíveis status da requisição é uma das preocupações que devemos ter. Por exemplo, o status 404 indica que o código da unidade de distribuição não existe.



```
1 package br.com.correios.exemplo;
2
3 import java.io.IOException;
4
15 public class Cliente {
16
17     public static void main(String[] args) throws ClientProtocolException, IOException {
18         String url = "http://app.correiosnet.int/geo/unidades/00057878";
19
20         HttpGet request = new HttpGet(url);
21         request.addHeader("Authorization", "Basic MTIzNDp0ZXN0ZQ==");
22
23         HttpClient client = new DefaultHttpClient();
24         HttpResponse response = client.execute(request);
25
26         int statusCode = response.getStatusLine().getStatusCode();
27
28         switch (statusCode) {
29             case 200:
30                 InputStream content = response.getEntity().getContent();
31
32                 Reader reader = new InputStreamReader(content);
33                 System.out.println("XML: " + CharStreams.toString(reader));
34                 break;
35             case 404:
36                 System.out.println("Recurso não encontrado...");
37                 break;
38             case 500:
39                 System.out.println("Erro interno do servidor...");
40                 break;
41             default:
42                 System.out.println("Falha ao executar o serviço...");
43                 break;
44         }
45     }
46 }
47
48 }
49
50 }
```

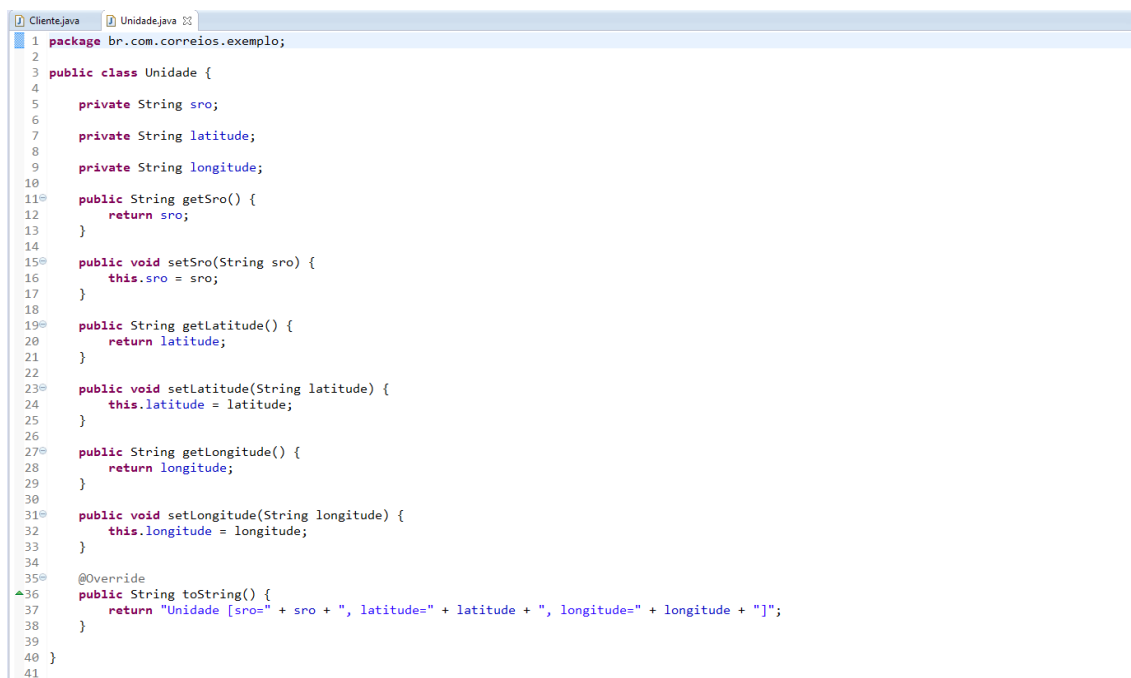
Imagem 10: Tratamento do status da requisição

### 7.3 Conversão

Lidar diretamente com a resposta da requisição não é uma boa solução. Portanto, é melhor convertê-la em uma estrutura de dados mais simples de ser manipulada. Em outras palavras, vamos extrair os dados do serviço de geografia das unidades de distribuição e armazená-los em um objeto.

Há várias formas de realizar essa conversão. Bibliotecas gratuitas e pagas estão disponíveis no mercado com essa finalidade, mas também é possível usar a própria API nativa do Java. Neste exemplo, usaremos a biblioteca XStream (<http://x-stream.github.io>) por ser simples e gratuita.

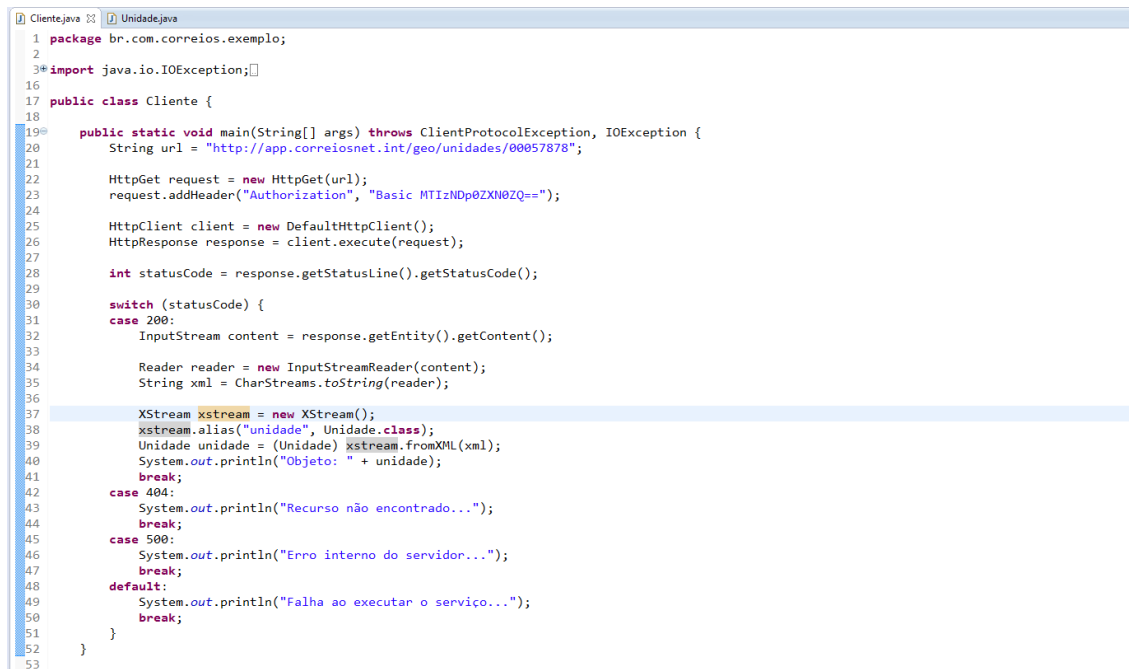
Crie um POJO (Plain Old Java Objects) para representar os dados do serviço de geografia das unidades de distribuição.



```
1 package br.com.correios.exemplo;
2
3 public class Unidade {
4
5     private String sro;
6
7     private String latitude;
8
9     private String longitude;
10
11     public String getSro() {
12         return sro;
13     }
14
15     public void setSro(String sro) {
16         this.sro = sro;
17     }
18
19     public String getLatitude() {
20         return latitude;
21     }
22
23     public void setLatitude(String latitude) {
24         this.latitude = latitude;
25     }
26
27     public String getLongitude() {
28         return longitude;
29     }
30
31     public void setLongitude(String longitude) {
32         this.longitude = longitude;
33     }
34
35     @Override
36     public String toString() {
37         return "Unidade [sro=" + sro + ", latitude=" + latitude + ", longitude=" + longitude + "];"
38     }
39
40 }
41
```

Imagem 11: POJO

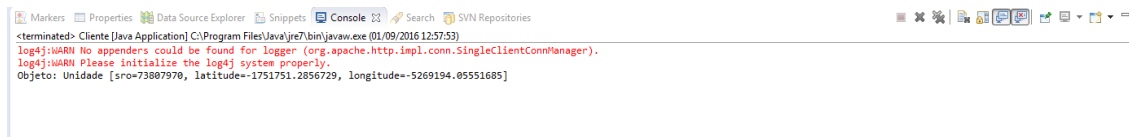
Em seguida, use a biblioteca XStream para converter a resposta da requisição em um objeto desse POJO.



```
1 package br.com.correios.exemplo;
2
3 import java.io.IOException;
4
16 public class Cliente {
17
18
19 public static void main(String[] args) throws ClientProtocolException, IOException {
20     String url = "http://app.correiosnet.int/geo/unidades/00057878";
21
22     HttpGet request = new HttpGet(url);
23     request.addHeader("Authorization", "Basic MTIzNDp0ZXN0ZQ==");
24
25     HttpClient client = new DefaultHttpClient();
26     HttpResponse response = client.execute(request);
27
28     int statusCode = response.getStatusLine().getStatusCode();
29
30     switch (statusCode) {
31     case 200:
32         InputStream content = response.getEntity().getContent();
33
34         Reader reader = new InputStreamReader(content);
35         String xml = CharStreams.toString(reader);
36
37         XStream xstream = new XStream();
38         xstream.alias("unidade", Unidade.class);
39         Unidade unidade = (Unidade) xstream.fromXML(xml);
40         System.out.println("Objeto: " + unidade);
41         break;
42     case 404:
43         System.out.println("Recurso não encontrado...");
44         break;
45     case 500:
46         System.out.println("Erro interno do servidor...");
47         break;
48     default:
49         System.out.println("Falha ao executar o serviço...");
50         break;
51     }
52 }
53 }
```

Imagem 12: Conversão

A partir desse momento, a manipulação dos dados do serviço se dará por meio desse objeto.



```
<terminated> Cliente [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (01/09/2016 12:57:53)
log4j:WARN No appenders could be found for logger [org.apache.http.impl.conn.SingleClientConnManager].
log4j:WARN Please initialize the log4j system properly.
Objeto: Unidade [sro=73807970, latitude=-1751751.2856729, longitude=-5269194.05551685]
```

Imagem 13: Objeto de manipulação

Este tutorial deste item apresentou a implementação de um cliente Java usando bibliotecas simples e gratuitas, para acesso aos dados do serviço de geografia das unidades de distribuição.

## **8. Dúvidas**

Em caso de dúvida, favor contatar a caixa postal: [correioslogti@correios.com.br](mailto:correioslogti@correios.com.br)