



# APIs e REST



© 2024 Amazon Web Services, Inc. ou suas afiliadas. Todos os direitos reservados.

1



## Principal objetivo da unidade

Você aprenderá a descrever o propósito e a função das APIs, incluindo a REST.



© 2024 Amazon Web Services, Inc. ou suas afiliadas. Todos os direitos reservados.

2

Você aprenderá a fazer o seguinte:

- Descrever a finalidade e a função das interfaces de programação de aplicativo (APIs), incluindo a Transferência de estado representacional (REST).



## O que é uma API?

- Uma API fornece acesso programático a uma aplicação.
  - A aplicação-cliente envia uma solicitação à aplicação do servidor usando a API da aplicação do servidor.
  - A aplicação do servidor retorna uma resposta à aplicação-cliente.
- Os benefícios de uma API incluem o seguinte:
  - Fornece acesso às funções de uma aplicação sem usar uma interface gráfica do usuário (GUI)
  - Fornece uma maneira consistente de invocar as funções de uma aplicação



A maioria das pessoas que usam computadores está familiarizada com o uso de uma interface gráfica do usuário (GUI) para interagir com uma aplicação. Por exemplo, o Console de Gerenciamento da AWS é uma interface gráfica que uma pessoa pode usar para interagir com os serviços da AWS.

Quando os programas de computador precisam se comunicar entre si, eles usam APIs. As APIs tornam o software de computador acessível aos desenvolvedores por meio de códigos, para que eles possam criar programas de software que interagem com outros programas de software. As APIs funcionam como intermediárias para que as aplicações possam se comunicar entre si. Essa comunicação é iniciada com uma chamada de API. Uma chamada de API é uma mensagem que é entregue a uma aplicação na forma de solicitação, depois é retornada como resposta ao solicitante. As APIs definem uma forma padrão de invocar as funções da aplicação de forma programática.

## Exemplo de API



Aplicativo móvel  
de meteorologia

API

Sistema de software da  
agência meteorológica



© 2024 Amazon Web Services, Inc. ou suas afiliadas. Todos os direitos reservados.

4

Por exemplo, o sistema de software da agência meteorológica contém dados meteorológicos diários. A aplicação de meteorologia do seu celular se comunica com esse sistema por meio de APIs e mostra atualizações meteorológicas diárias nele.

As APIs públicas dos serviços da AWS são outro exemplo de API. Quando você usa uma ferramenta, como a AWS Command Line Interface (AWS CLI), a ferramenta usa as APIs públicas expostas pelos serviços da AWS para invocar funções de serviço.

## O que são as APIs RESTful?

- Uma API RESTful é uma interface que dois sistemas de computador usam para trocar informações com segurança pela internet.
  - Projetada para aplicações baseadas em rede e fracamente acopladas
  - Comunica-se por HTTP
  - Expõe recursos em URLs específicas
- Muitas APIs da web são RESTful.



Algumas APIs seguem os princípios da REST. As APIs REST se comunicam por HTTP e seguem padrões de comunicação de software seguros, confiáveis e eficientes. REST é um design de API muito conhecido que substituiu amplamente o Simple Object Access Protocol (SOAP) como padrão para serviços da web. As APIs baseadas na web que seguem os princípios de design REST são chamadas de RESTful.

## Princípios de design da API RESTful



© 2024 Amazon Web Services, Inc. ou suas afiliadas. Todos os direitos reservados.

6

A REST segue cinco princípios de design:

- O cliente deve ter uma interface uniforme para acessar o servidor. Uma solicitação deve ser feita para um único endpoint ou URI ao interagir com cada recurso distinto que faz parte do serviço. A interface não define a estrutura do que é retornado quando o cliente faz uma solicitação.
- Os serviços RESTful também são stateless. Stateless significa que o servidor não monitora quais solicitações o cliente da conexão fez ao longo do tempo. Significa ainda que ele não acompanha qual etapa o cliente pode ter concluído com relação a uma série de ações. Na verdade, todas as informações da sessão sobre o cliente são conhecidas apenas pelo cliente. O servidor não conhece o estado do cliente e o cliente não conhece o estado do servidor.
- Os clientes REST devem poder armazenar em cache as respostas recebidas do servidor REST.
- Os serviços RESTful comportam sistemas em camadas, o que significa que o cliente pode se



A REST segue cinco princípios de design:

- O cliente deve ter uma interface uniforme para acessar o servidor. Uma solicitação deve ser feita para um único endpoint ou URI ao interagir com cada recurso distinto que faz parte do serviço. A interface não define a estrutura do que é retornado quando o cliente faz uma solicitação.
- Os serviços RESTful também são stateless. Stateless significa que o servidor não monitora quais solicitações o cliente da conexão fez ao longo do tempo. Significa ainda que ele não acompanha qual etapa o cliente pode ter concluído com relação a uma série de ações. Na verdade, todas as informações da sessão sobre o cliente são conhecidas apenas pelo cliente. O servidor não conhece o estado do cliente e o cliente não conhece o estado do servidor.
- Os clientes REST devem poder armazenar em cache as respostas recebidas do servidor REST.
- Os serviços RESTful comportam sistemas em camadas, o que significa que o cliente pode se conectar a um servidor intermediário. O servidor REST pode ser distribuído, o que oferece suporte ao balanceamento de carga.
- Opcionalmente, os serviços RESTful podem comportar código sob demanda. Código sob demanda significa que o servidor pode passar um código (que pode ser executado) para o cliente, como algum JavaScript. Esse recurso estende a funcionalidade do cliente REST.

OBSERVAÇÃO: o uso de código sob demanda não é amplamente adotado entre os desenvolvedores e ainda é um componente opcional do desenvolvimento de APIs.

## Componentes da RESTful

Veja a seguir os componentes da RESTful:

- Cliente
- Recurso
- Solicitação
- Resposta



© 2024 Amazon Web Services, Inc. ou suas afiliadas. Todos os direitos reservados.

7

Veja a seguir os componentes da RESTful:

- **Cliente:** os clientes são usuários que desejam acessar informações da web. O cliente pode ser uma pessoa ou um sistema de software que usa a API. Por exemplo, os desenvolvedores podem escrever programas que acessam dados meteorológicos de um sistema de meteorologia. Você também pode acessar os mesmos dados do seu navegador ao visitar diretamente o site de meteorologia.
- **Recurso:** os recursos são as informações que diferentes aplicações fornecem a seus clientes. Os recursos podem ser imagens, vídeos, texto, números ou qualquer tipo de dado. A máquina que fornece o recurso ao cliente também é chamada de servidor. As organizações usam APIs para compartilhar recursos e fornecer serviços da web, mantendo a segurança, o controle e a autenticação. Além disso, as APIs os ajudam a determinar quais clientes têm acesso a recursos internos específicos.





Veja a seguir os componentes da RESTful:

- **Cliente:** os clientes são usuários que desejam acessar informações da web. O cliente pode ser uma pessoa ou um sistema de software que usa a API. Por exemplo, os desenvolvedores podem escrever programas que acessam dados meteorológicos de um sistema de meteorologia. Você também pode acessar os mesmos dados do seu navegador ao visitar diretamente o site de meteorologia.
- **Recurso:** os recursos são as informações que diferentes aplicações fornecem a seus clientes. Os recursos podem ser imagens, vídeos, texto, números ou qualquer tipo de dado. A máquina que fornece o recurso ao cliente também é chamada de servidor. As organizações usam APIs para compartilhar recursos e fornecer serviços da web, mantendo a segurança, o controle e a autenticação. Além disso, as APIs os ajudam a determinar quais clientes têm acesso a recursos internos específicos.
- **Solicitação:** as solicitações são enviadas pelo cliente para um servidor. A solicitação é formatada de modo que o servidor consiga entendê-la.
- **Resposta:** a resposta é o que o servidor envia de volta para responder à solicitação do cliente. Essa informação inclui uma mensagem de status de sucesso ou falha, um corpo de mensagem que contém a representação do recurso e metadados sobre a resposta.

## Formato de solicitação REST

Uma solicitação REST inclui o seguinte:

- Endpoint (como um URL)
- Método
  - GET: para ler um recurso
  - POST: para criar um recurso
  - PUT: para atualizar um recurso existente
  - DELETE: para excluir um recurso
- Cabeçalho
- Corpo (dados)



© 2024 Amazon Web Services, Inc. ou suas afiliadas. Todos os direitos reservados.

8

Uma solicitação REST inclui vários componentes:

- **Endpoint:** quando você faz uma solicitação para um servidor REST, precisa conhecer o endpoint. O formato do endpoint é um URL. O URL permite que o cliente notifique o servidor sobre os recursos com os quais deseja interagir.
- **Método:** cada recurso exposto na API REST comporta um ou mais métodos, que incluem os seguintes:
  - GET: o servidor é solicitado a recuperar um recurso.
  - POST: o servidor é solicitado a criar um recurso.
  - PUT: o servidor é solicitado a editar ou atualizar um recurso existente.
  - DELETE: o servidor é solicitado a excluir um recurso.
- **Cabeçalho:** o cabeçalho contém os metadados da solicitação.
- **Corpo da solicitação:** o corpo de uma solicitação apresenta os dados que o cliente envia ao

Uma solicitação REST inclui vários componentes:

- **Endpoint:** quando você faz uma solicitação para um servidor REST, precisa conhecer o endpoint. O formato do endpoint é um URL. O URL permite que o cliente notifique o servidor sobre os recursos com os quais deseja interagir.
- **Método:** cada recurso exposto na API REST comporta um ou mais métodos, que incluem os seguintes:
  - **GET:** o servidor é solicitado a recuperar um recurso.
  - **POST:** o servidor é solicitado a criar um recurso.
  - **PUT:** o servidor é solicitado a editar ou atualizar um recurso existente.
  - **DELETE:** o servidor é solicitado a excluir um recurso.
- **Cabeçalho:** o cabeçalho contém os metadados da solicitação.
- **Corpo da solicitação:** o corpo de uma solicitação apresenta os dados que o cliente envia ao servidor. Uma solicitação POST ou PUT normalmente contém um corpo. Por exemplo, se você invocar um método PUT, é bem provável que precise incluir detalhes na solicitação que indiquem como um recurso existente deve ser atualizado. No entanto, as solicitações GET raramente incluem um corpo.

## Exemplo de solicitação RESTful

Considere o exemplo da criação de um bucket chamado pets no Amazon Simple Storage Service (Amazon S3).



© 2024 Amazon Web Services, Inc. ou suas afiliadas. Todos os direitos reservados.

9

Considere criar um bucket chamado pets no Amazon Simple Storage Service (Amazon S3). Dê uma olhada no código. Você deve lembrar que o método PUT API é usado principalmente para atualizar recursos existentes e também pode ser usado para criar um recurso.

- Nome do bucket: pets
- Endpoints para o Amazon S3 com a seguinte estrutura: `s3.<Region>.amazonaws.com`
- Content-Length: tamanho da mensagem (sem os cabeçalhos)
  - No exemplo, o comprimento do conteúdo é 0 e o URI do recurso está em branco (/) porque o cliente não precisa enviar nada ao servidor para criar o bucket.
- Date: a data e a hora da solicitação
- Authorization: as informações de credencial para autenticar a solicitação no servidor

## Exemplo de resposta RESTful

Veja a seguir a resposta retornada pela API createBucket do Amazon S3.

```
HTTP/1.1 200 OK
x-amz-id-2: YgIPiFbika2bj0KMg95r/0zo3emzU4dzsD4rcKCHQUAdQkf3ShJT00pXUueF6QKo
x-amz-request-id: 236A8905248E5A01
Date: Wed, 01 Mar 2006 12:00:00 GMT
Location: /pets
Content-Length: 0
Connection: close
Server: AmazonS3
```

Diagram illustrating the components of the response:

- Código de status HTTP**: Points to the status code 200 OK.
- Token x-amz-id-2**: Points to the x-amz-id-2 header value.
- Data, localização, comprimento do conteúdo, conexão e servidor**: Points to the Date, Location, Content-Length, Connection, and Server headers.



© 2024 Amazon Web Services, Inc. ou suas afiliadas. Todos os direitos reservados.

10

Reserve um momento para examinar o cabeçalho da resposta. As informações a seguir fornecem detalhes sobre o cabeçalho da resposta.

A resposta mostra o seguinte:

- Código de status HTTP de 200 OK: Indicação de sucesso. O servidor recebeu e aceitou a solicitação.
- x-amz-id-2: token especial usado com o cabeçalho x-amz-request-id para ajudar a AWS a solucionar problemas.
- Date: data e hora em que o Amazon S3 respondeu; por exemplo, Wed, 01 Mar 2006 12:00:00 GMT.
- Location: a localização de um recurso recém-criado no Amazon S3 (nome do bucket).
- Content-Length: o tamanho em bytes do corpo na resposta.
- Connection: informações sobre se a conexão com o servidor está aberta ou fechada.
- Server: o nome do servidor que criou a resposta.

## Exemplo de REST com cURL

**Solicitação:**

```
Curl -I
-X POST
-d @file.json
-H "Content-Type: application/json"
https://www.example.com/someresource/
```



Cliente

HTTPS



Interface  
de API



Servidor REST

**Resposta:**

```
{
  "Status" : "Accepted"
}
```



© 2024 Amazon Web Services, Inc. ou suas afiliadas. Todos os direitos reservados.

11

O URL do cliente (cURL) é uma ferramenta de linha de comando popular que você pode usar para obter ou enviar arquivos via HTTP(S). Essa ferramenta é útil para testar o acesso à API REST. Neste exemplo, o usuário está criando um arquivo em um servidor.

No exemplo, observe o seguinte:

- `-i` indica que a resposta deve incluir os cabeçalhos de resposta.
- `-X` indica o método que você está invocando; por exemplo, POST.
- `-d` costuma ser usado com os métodos POST ou PUT e fornece os dados (corpo) que estão sendo enviados para o servidor REST. Nesse caso, os dados estão em um arquivo chamado file.json.
- `-H` transmite as informações do cabeçalho. O cabeçalho pode conter informações, como o tipo de conteúdo, que uma resposta aceitável incluirá.

Uma solicitação também deve fornecer as credenciais para a autenticação no cabeçalho da solicitação. Esse exemplo não inclui credenciais de autenticação.

## Códigos de status HTTP

Grupo de códigos de status	Descrição	Exemplos
1xx	Resposta informativa	100: indica o código da resposta informativa de que tudo está OK até agora e que o cliente deve prosseguir com a solicitação.
2xx	Sucesso	200: indica êxito. O servidor recebeu e aceitou a solicitação.
3xx	Redirecionamento	307: indica que o recurso de destino reside temporariamente em um URI diferente.
4xx	Erros do cliente	401: indica um erro do cliente. Não autorizado. A autenticação é necessária, mas as credenciais fornecidas não foram aceitas, ou talvez nenhuma credencial tenha sido fornecida na solicitação.
5xx	Erros do servidor	503: indica que o serviço não está disponível. Tente de novo mais tarde.



© 2024 Amazon Web Services, Inc. ou suas afiliadas. Todos os direitos reservados.

12

Um código de status HTTP é incluído nas respostas da API REST. É útil saber o que os códigos significam, pois eles indicam se a solicitação foi recebida com êxito ou se houve erros. Entender o que os códigos de status indicam, especialmente os de erro, pode ser útil ao solucionar problemas com as solicitações de API REST.

Este slide inclui os grupos de códigos de status para sua referência.

Existem centenas de códigos de status, mas eles são agrupados para que o primeiro dos três dígitos indique a natureza geral do status.

**\*\*Para acessibilidade:** tabela de códigos de status HTTP. A primeira coluna fornece os grupos de códigos de status. A segunda coluna descreve o código de status. A terceira coluna apresenta um exemplo. **Fim da descrição.**





## Perguntas de verificação

1. Para que é usada uma API?
2. Qual protocolo os serviços da web RESTful usam?
3. O que significa o código de status 200 nas respostas da API REST?



As respostas das perguntas são estas:

1. Para que é usada uma API?  
Você pode usar as APIs para interagir de maneira programática com uma aplicação ou um software separado.
2. Qual protocolo os serviços da web RESTful usam?  
Os serviços da web RESTful usam o protocolo HTTP. Os métodos padrão HTTP são usados para acessar recursos em uma arquitetura RESTful.
3. O que significa o código de status 200 nas respostas da API REST?  
Um código de status 200 indica sucesso. O servidor recebeu e aceitou a solicitação.





## Ideias principais



- As APIs fornecem acesso programático a uma aplicação e geralmente são usadas para programas que se comunicam entre si.
- Os princípios de design da REST são interface uniforme, condição stateless, capacidade de armazenamento em cache, sistema em camadas e, opcionalmente, código sob demanda.
- Os componentes de uma solicitação REST são endpoint (URL), método, cabeçalho e corpo.
- Os códigos de status HTTP podem ser úteis para solucionar erros da API REST.



## Agradecemos sua atenção

Correções, feedback ou outras dúvidas?

Entre em contato conosco em

<https://support.aws.amazon.com/#/contacts/aws-training>.

Todas as marcas comerciais pertencem aos respectivos proprietários.

© 2024 Amazon Web Services, Inc. ou suas afiliadas. Todos os direitos reservados.

15