忙しい人のための

Web APIとHTTP講座

V1.0　2020/1/29

覚えてほしいこと

1. Web APIとは何か、また、その代表的な実装方式。
2. HTTPのリクエストとレスポンス、その主な構成要素。
3. HTTPの代表的なメソッド。

そもそもAPIって何？

コンピュータプログラムの提供する機能を外部の別のプログラムから呼び出して利用するための手順・規約のこと。

Web APIって何？

HTTP(プロトコル)の仕組みを最大限利用したAPIのこと。
実装方式にはいくつかの種類がある。

Web APIの代表的な実装方式

- SOAP
  - Simple Object Access Protocol の略だが、全くシンプルではない。
  - XMLで通信の規約(WSDL)を取り決め、XMLでデータをやり取りする。重量級。

- REST
  - REpresentational State Transferの略だが、イメージしづらい。
  - HTTPのメソッドに則り、(主に)JSONでデータをやり取りする。軽量級。
  - 最近ではAPI = Web API = REST APIといっていいほど主流。
  - しっかりと原則を守ったRESTな仕組みをRESTfulと呼ぶ。

JSONって何？

JavaScript Object Notationの略でデータフォーマットの一種。人間にも機械にもわかりやすく、かつ余計な文字が少ないため軽いという素敵なフォーマット。

```
1. {
2.     "user": "太郎",
3.     "age": 23,
4.     "gender": "男"
5. }
```

参考：SOAPのWSDL

とても重厚な定義が必要…。

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions

  <!--
    対象名前空間URIの定義。よくわからんけど任意の値で良いみたい。
    多くのサンプルではそのWEBサービスのURIを記載していた。
  -->
  targetNamespace="http://example.com/Service01"

  <!--
    わからんけど親がtargetNamespaceなのに対して、子がxmlns:implみたいな印象を受けた。
    任意の値でいいっぽい。
  -->
  xmlns:impl="http://example.com/Service01"

  <!--
    xmlnsは、各々で定義されてるxmlマークアップを使うということ。おまじない程度の理解で良さそ
  -->
  xmlns:apachesoap="http://xml.apache.org/xml-soap"
  xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
  xmlns="http://schemas.xmlsoap.org/wsdl/">

  <!--
    WEBサービスで使用されるmessage(インタフェース？)の抽象的定義。
  -->
  <wsdl:message name="addHexString">
    <!-- APIパラメータ名と型の定義 -->
    <wsdl:part name="text" type="xsd:string" />
    <wsdl:part name="num" type="xsd:int" />
  </wsdl:message>
```

～

```
    <wsdlsoap:binding style="rpc"
            transport="http://schemas.xmlsoap.org/soap/http" />
      <!-- portTypeで定義した内容をさらに具体的に定義？ -->
      <wsdl:operation name="addHexString">
        <wsdlsoap:operation soapAction=""/>
        <wsdl:input name="addHexString">
          <!-- SOAPメッセージ本体だとか。なんだよ本体って。 -->
          <wsdlsoap:body use="encoded"
              encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
        </wsdl:input>
        <wsdl:output name="addHexStringResponse">
          <wsdlsoap:body use="encoded"
              encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
        </wsdl:output>
      </wsdl:operation>
    </wsdl:binding>

    <!-- 上のportType要素で定義したポートのうち、関連するポートをひとまとめにする。 -->
    <wsdl:service name="Service01">
      <!--
        nameはportTypeで定義した名前。
        bindingは対応するbindingの名前。
      -->
      <wsdl:port name="Service01"
          binding="impl:Service01SoapBinding">
        <!-- 具体的なアドレスを指定 -->
        <wsdlsoap:address
          location="http://mac-book-air.local:8088/mockService01SoapBinding"/>
      </wsdl:port>
    </wsdl:service>
</wsdl:definitions>
```

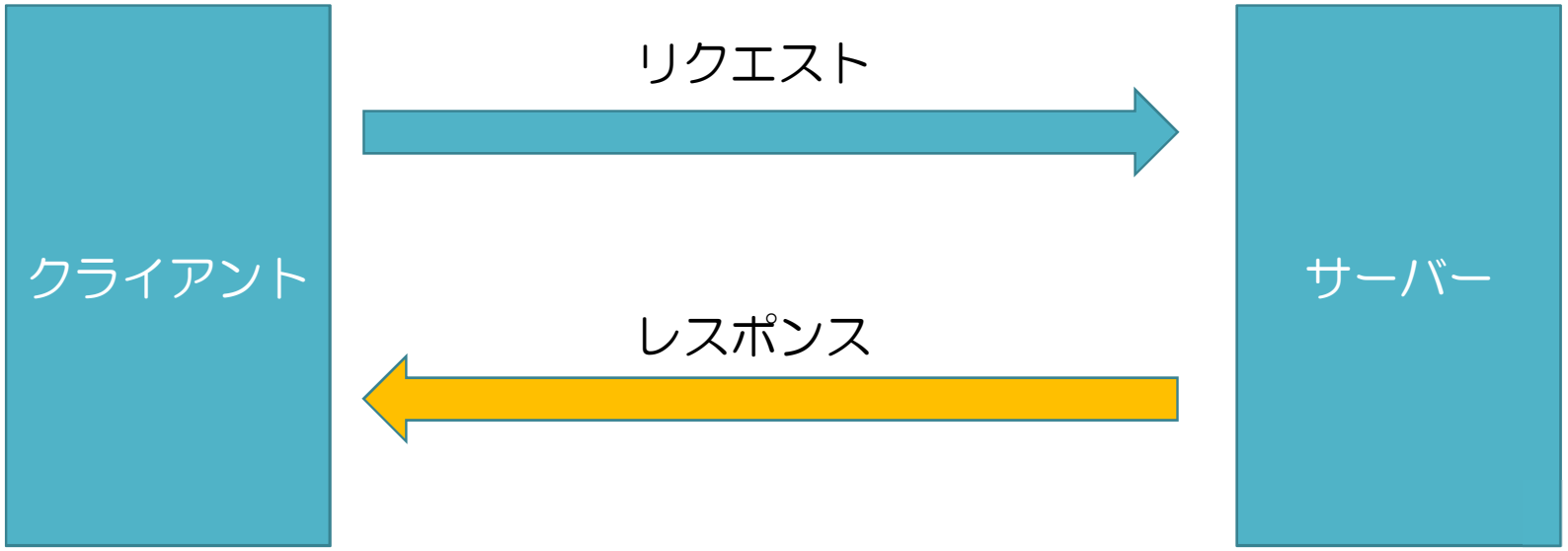
HTTP超概要

HTTPとは？

サーバーとクライアント間で情報を送受信するための決まり事を定義したもの（＝プロトコル）

HTTP通信とは？

HTTPの決まり事に従い、指定したサーバーのリソースを取りに行く（または渡す）クライアントとサーバーのステートレスなやり取り。クライアントからの要求をリクエスト、サーバーからの返答をレスポンスと呼び、それぞれ様々な情報を詰めてやり取りする。



URL/URI

URL　：Uniform Resource Locator

URI　：Uniform Resource Identifier

URL　：Web上にあるあらゆるリソースがWeb上のどの位置にあるのかを表したもの。

URI　：Web上にあるあらゆるリソースを一意に表すための識別子。

URL≒URI

Web APIの世界では（言葉として）URIを使うのが主流。

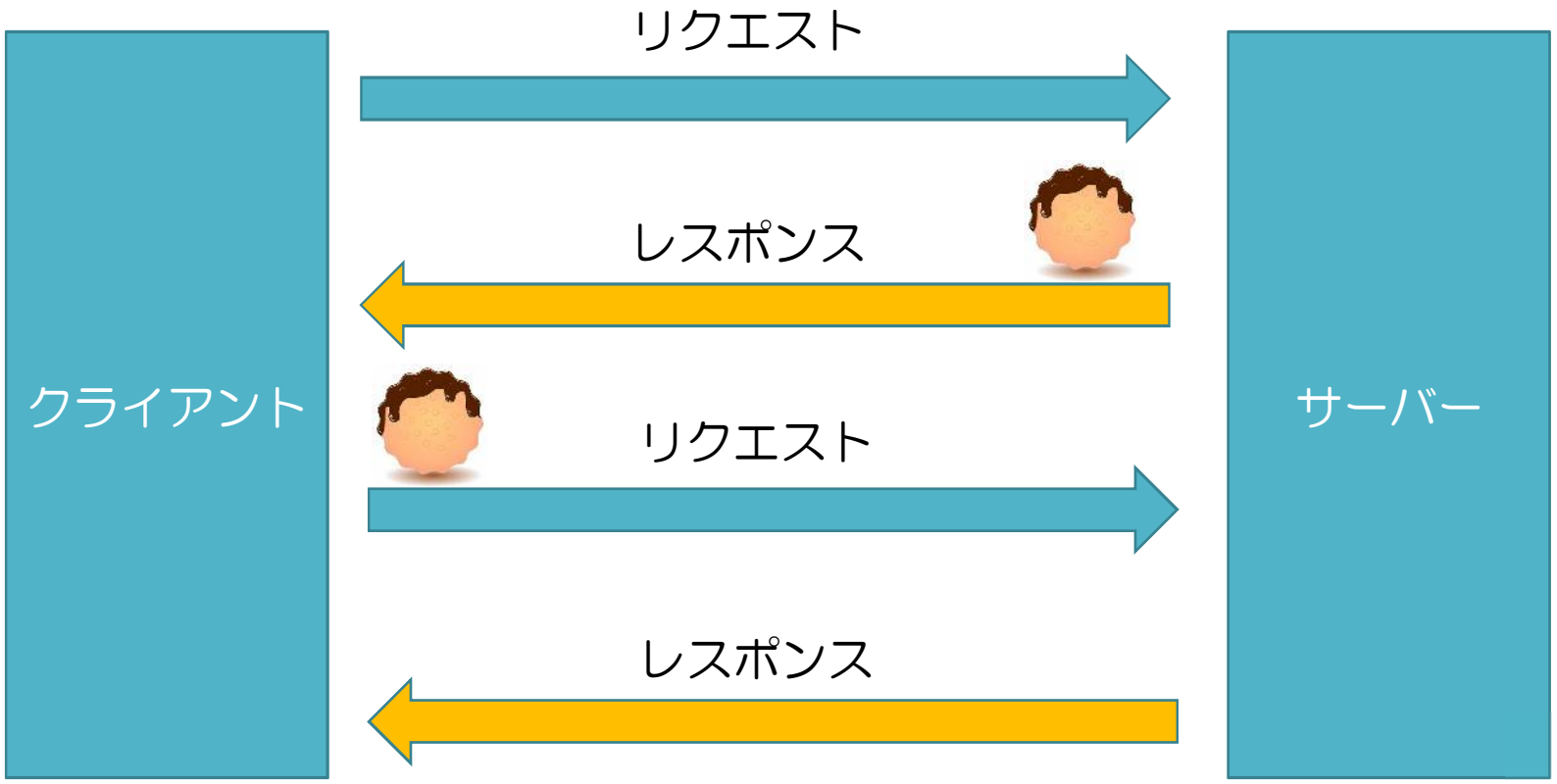
Web APIにアクセスするためのURIをエンドポイントとも呼ぶ。

ステートレスとは？

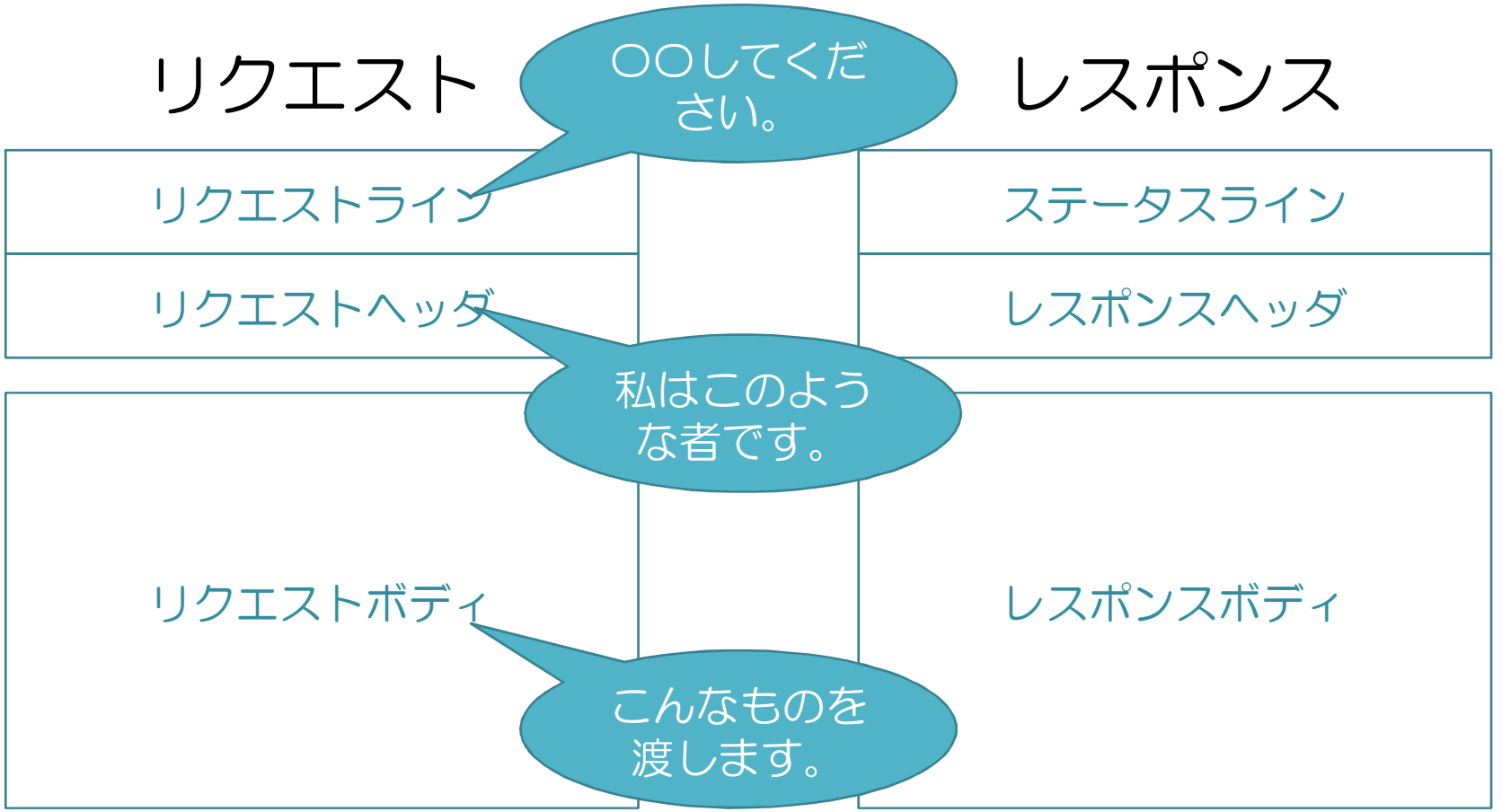
ステートレス：状態を維持しない。
HTTP自身には以前送られてきたリクエストや送ったレスポンスについて記憶する仕組みがない。

状態を維持するには？

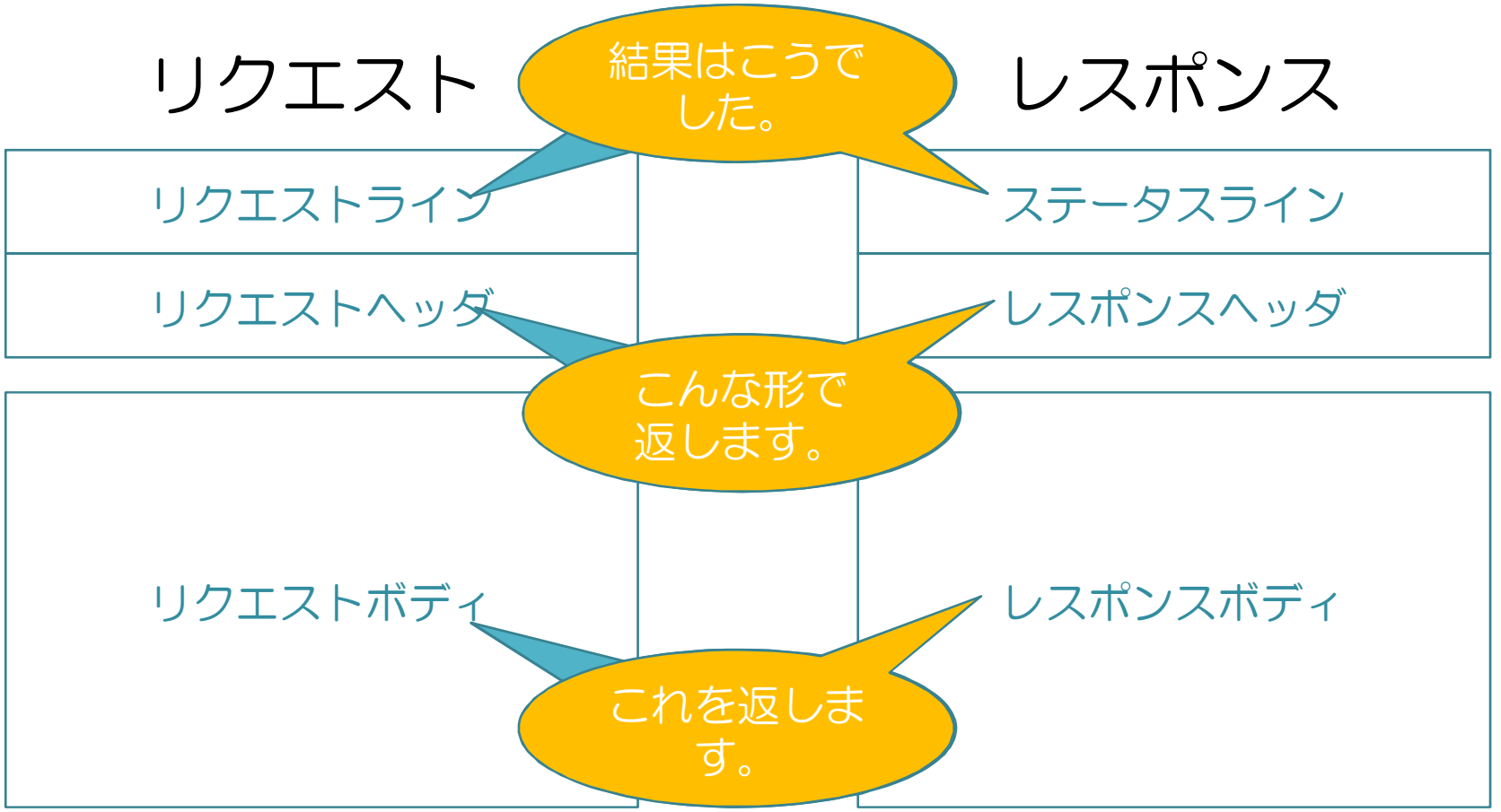
Cookieという仕組みを使い、クライアントを認識する。



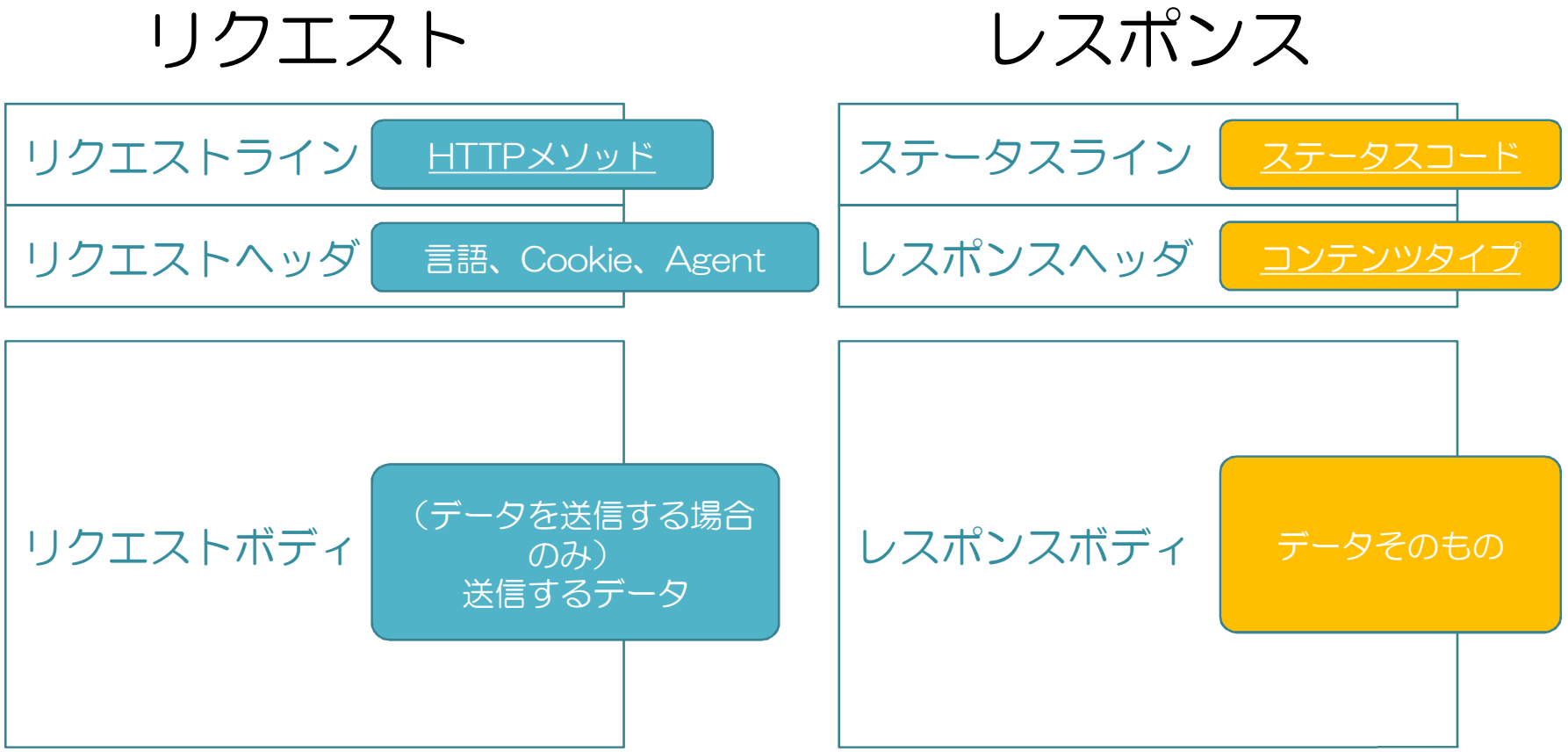
リクエストとレスポンスの中身



リクエストとレスポンスの中身



リクエストとレスポンスの中身



※主なもののみ抜粋

基本のHTTPメソッド

メソッド	説明
GET	リソースの取得
POST	リソースの(新規登録のための)送信

その他のHTTPメソッド

メソッド	説明
PUT	リソースの更新
DELETE	リソースの削除

など

リクエストヘッダー

パラメータ名	例
Accept-Language	ja; en-US;
Cookie	_gid=GA1.3.12892389;
Referer	https://www.google.com/
User-Agent	Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.130 Safari/537.36

など

リクエストボディ

```
<form action="/Auth" method="post" />
        <input id="Id" type="text"/>
        <input id="Password" type="password" />
</form>
```

パラメータ名	値
Id	k_nakamura
Password	passw0rd!

ステータスコード（大枠）

ステータスコード	説明
200番台	リクエスト成功
400番台	クライアント側に起因するエラー
500番台	サーバ側に起因するエラー

など

代表的なステータスコード

ステータスコード	説明
200	リクエストに成功
404	指定されたリソースがない
500	サーバ側でエラーが発生した

など

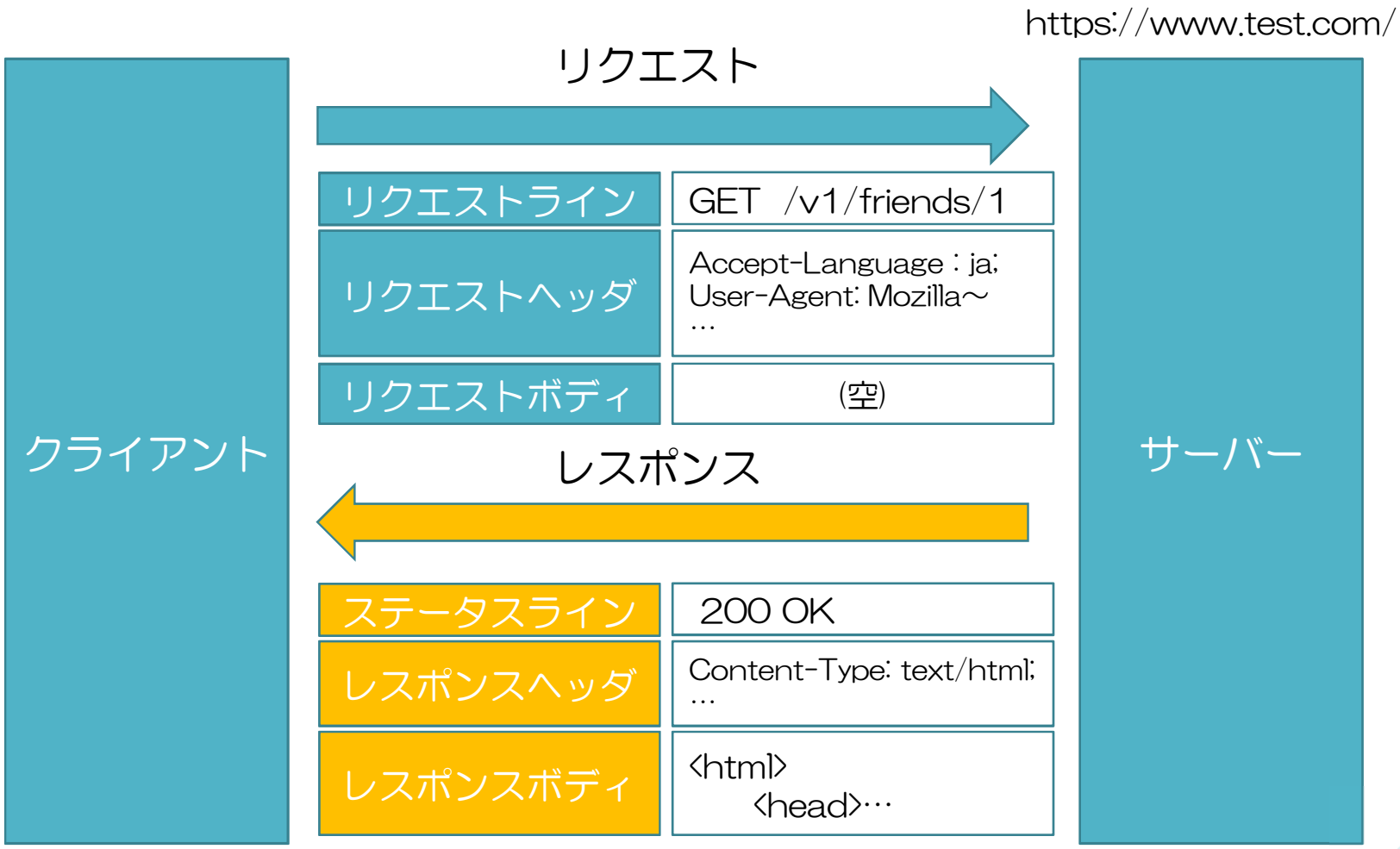
コンテンツタイプ

≒メディアタイプ、MIMEタイプ

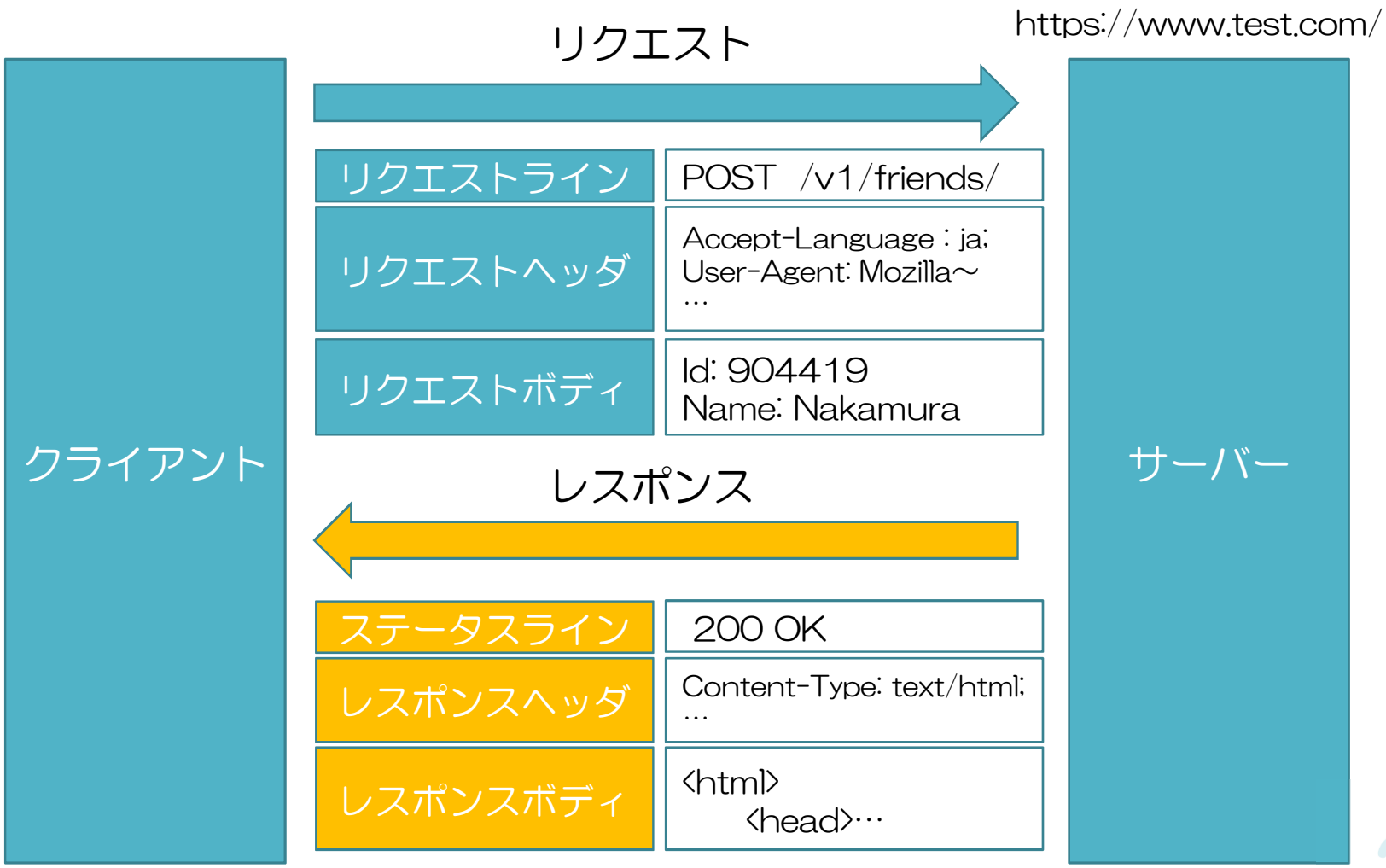
コンテンツタイプ	データ形式
text/plain	プレーンなテキスト
text/html	HTML文書
application/xml	XML文書
application/json	JSON文書

など

HTTP通信（GETの例）



HTTP通信（POSTの例）



まとめ

HTTPとは、あらゆるリソースを決め事に従い通信し、やり取りする仕組み。

リクエスト先	レスポンス
• ホームページ • Webアプリ	HTMLやCSS
• (REST)Web API	JSON

ホームページやWebアプリケーションはクライアントが(ほぼ)Webブラウザだが、Web APIはHTTPで通信が可能なあらゆるものがクライアントになる。

REST APIとHTTPの関係

（再掲）REST： HTTPのメソッドに則り、(主に)JSONでデータをやり取りする。

処理	HTTPメソッド	CRUD操作
取得	GET	READ
新規登録	POST	CREATE
更新	PUT	UPDATE
削除	DELETE	DELETE

REST APIのURI例

処理	HTTPメソッド	URI（エンドポイント）例
単一取得	GET	https://www.apitest.com/v1/friends/1
一覧取得	GET	https://www.apitest.com/v1/friends/
新規登録	POST	https://www.apitest.com/v1/friends/
更新	PUT	https://www.apitest.com/v1/friends/1
削除	DELETE	https://www.apitest.com/v1/friends/1

振り返り

1. Web APIの代表的な実装方式は何ですか？
2. HTTPのリクエストとレスポンス主な構成要素は何ですか？
3. HTTPの代表的なメソッドは何ですか？

おわり