

HTTP com BaseX

- O BaseX não é só uma ferramenta standalone mas também uma aplicação que suporta o protocolo **HTTP**;
- Na pasta de instalação do BaseX, mais especificamente na pasta bin, podemos encontrar os scripts: **basexhttp** (para sistemas unix) e **basexhttp.bat** (para windows);
- A execução do script respectivo irá iniciar um **servidor** (na porta 1984) e um servidor HTTP (na porta 8184 ou 8080).

- Verifique que o servidor HTTP está a executar;
- Para isso, abra o seu browser ou **Postman** e digite: <http://localhost:8984/> ou <http://localhost:8080/> ;

BaseX HTTP Services

Welcome to the BaseX HTTP Services. They allow you to:

- create web applications and services with [RESTXQ](#),
- use full-duplex communication with [WebSockets](#),
- query and modify databases via [REST](#) (try [here](#)), and
- browse and update resources via [WebDAV](#).

Find more information on the [Web Application](#) page in our documentation.

The following sample applications give you a glimpse of how applications can be written with BaseX:

[DBA: Database Administration](#)

The Database Administration interface is completely written in RESTXQ.

The source code helps to understand how complex web applications can be built with XQuery.

[WebSocket Chat](#)

The chat application demonstrates how bidirectional communication is realized with BaseX.

For a better experience when testing the chat, consider the following steps:

1. Create different database users first (e.g. via the DBA).
2. Open two different browsers and log in with different users.

O HTTP Client Module:

- https://docs.basex.org/wiki/HTTP_Client_Module ;
- Este módulo XQuery contém uma única função para enviar solicitações HTTP e manipular respostas HTTP.

```
let $req := http:send-request(<http:request method='get' 1  
                                json='format=xquery,lax=true' />, 2  
                                "https://www.amiiboapi.com/api/amiibo/?name=mario") 3  
return $req[2] 4
```

1- Configuração do método HTTP

2- As respostas CSV, JSON e HTML são automaticamente convertidas em uma representação XML.

O formato de destino pode ser alterado pelo fornecimento de atributos csv, json e html.

3- URL do pedido

4- A resposta devolve duas partes: dados da resposta (HTTP) e a resposta

Exemplo com um pedido com o método POST com cabeçalhos específicos e com body:

```
let $json-data := '{ (...) }'  
http:send-request (<http:request method='post' > 1  
    <http:header name="api-key" value="..." /> 2  
    <http:body media-type='application/json' /> 3  
    </http:request> ,  
    "https://data.mongodb-api.com...", $json-data) 4
```

- 1- Configuração do método POST
- 2- Header com api-key (utilizado para autorização)
- 3- Mime Type do conteúdo enviado
- 4- Conteúdo enviado no Body.

- O módulo **RESTXQ** (<http://docs.basex.org/wiki/RESTXQ>) proporciona a utilização do XQuery como uma linguagem server-side para a web;
- O modulo RESTXQ define um conjunto de anotações pré-definidas que permitem o mapeamento de pedidos HTTP em funções **XQuery** que geram e retornam respostas HTTP.

- O serviço **RESTXQ** é acessível através do URL: `http://localhost:8984/`;
- As **anotações** são definidas ao namespace: `http://exquery.org/ns/restxq`;
- Uma **Resource Function** é uma função XQuery que foi anotada com as anotações RESTXQ;
- Quando um pedido HTTP é enviado, a **Resource Function** é invocada de acordo com os critérios definidos pelas anotações.

Criar um endpoint com BaseX

- Exemplo Método GET:

```
module namespace page = 'http://basex.org/examples/web-page' ;
```

namespace

```
declare
  %rest:path("hello/{ $who }")
  %rest:GET
function page:hello( $who ) {
  <response>
    <title>Hello { $who }!</title>
  </response>
};
```

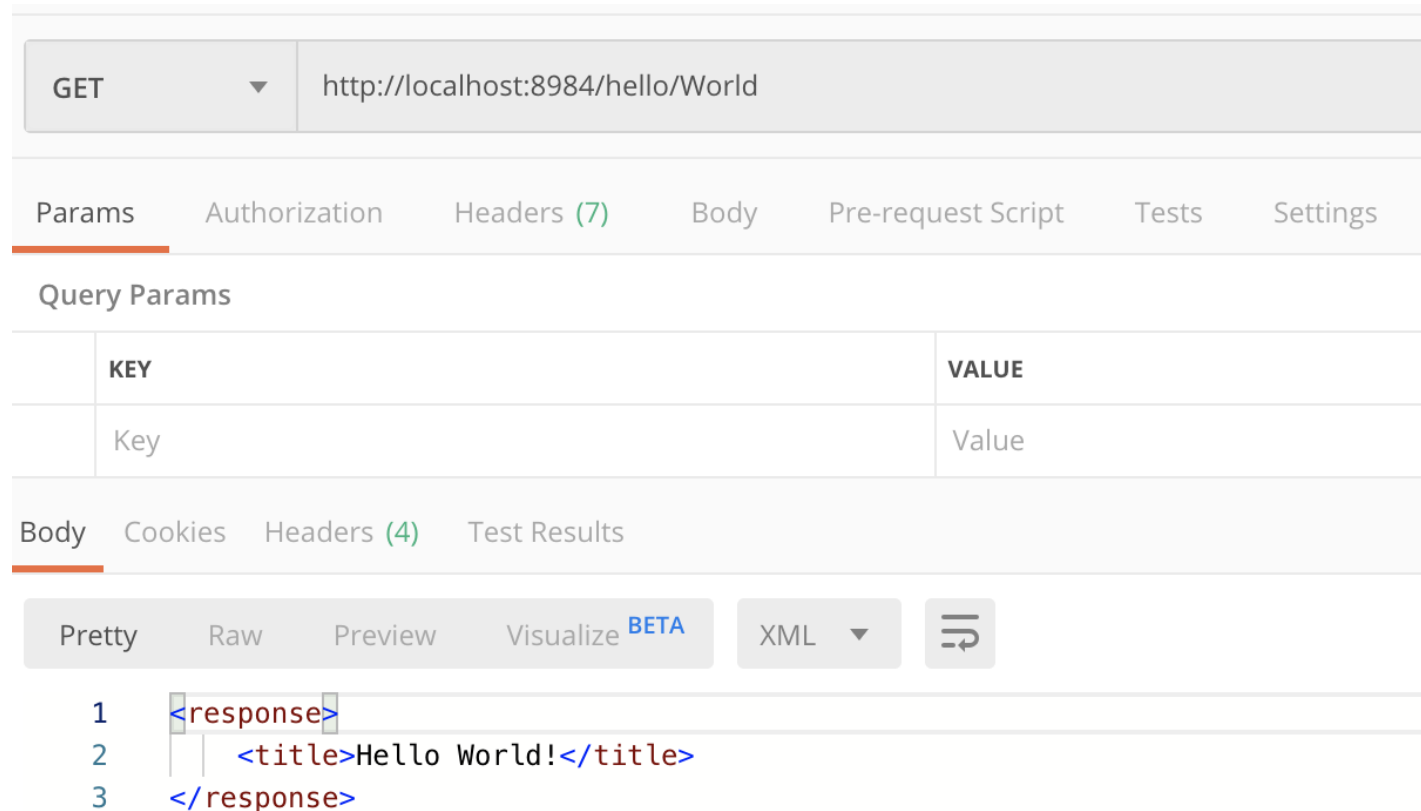
Anotações RESTXQ:

- A anotação path define o caminho para o [endpoint](#);
- GET indica o [método de requisição](#) de pedido HTTP aceite;

Função XQuery

Criar um endpoint com BaseX

Exemplo de invocação utilizando Postman:



Criar um endpoint com BaseX

Exemplo Método POST:

```
declare
  %rest:path("/form")
  %rest:POST
  %rest:form-param("message","{$message}", "(no message)")
  %rest:header-param("User-Agent", "{$agent}")
function page:hello-postman($message as xs:string, $agent as xs:string)
  as element(response) {
  <response type='form'>
    <message>{ $message }</message>
    <user-agent>{ $agent }</user-agent>
  </response>
};
```

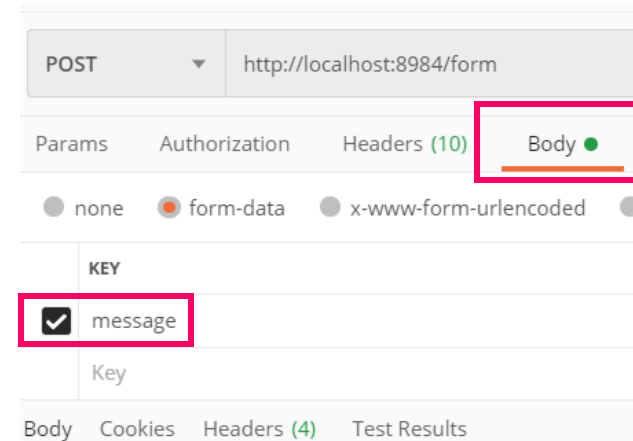
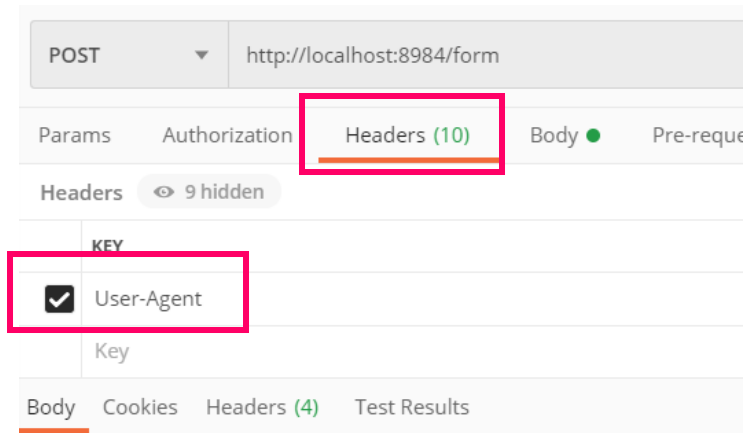
Conteúdo aceite deverá ser enviado no body como **form-data**

Conteúdo do cabeçalho!

Deve retornar um elemento: **response**

Criar um endpoint com BaseX

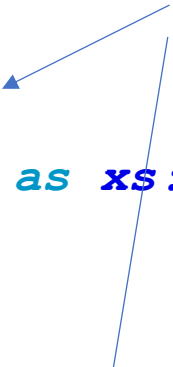
- Exemplo de invocação utilizando Postman:



Criar um endpoint com BaseX

- POST/PUT

```
declare
  %rest:path("/post")
  %rest:POST("{ $body }")
function page:post($body as xs:string) as xs:string {
  let $b := $body
  return $b
};
```



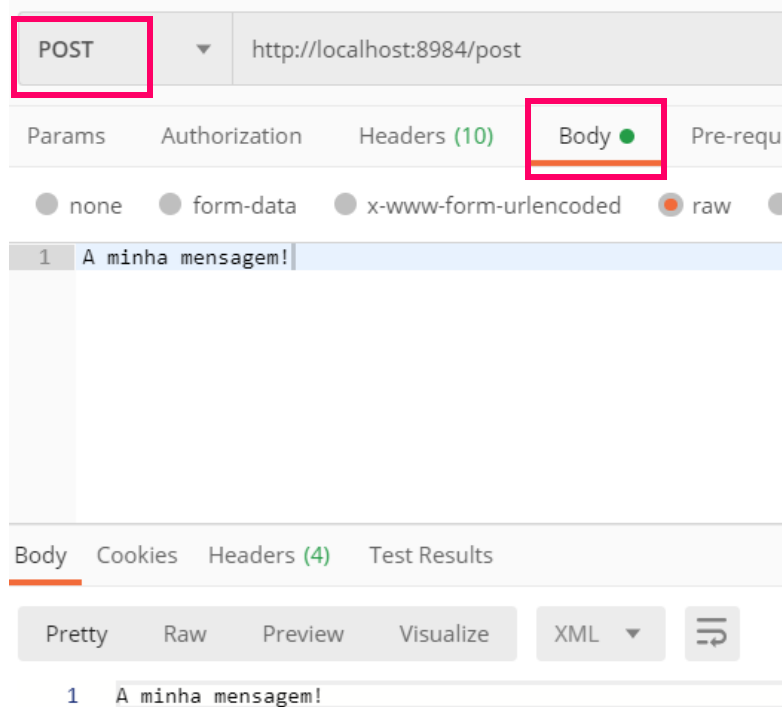
Envio de dados através do
corpo do pedido

```
declare %rest:PUT("{ $body }")
  %rest:path("/put")
function page:put($body) {
  "Request body: " || $body
};
```

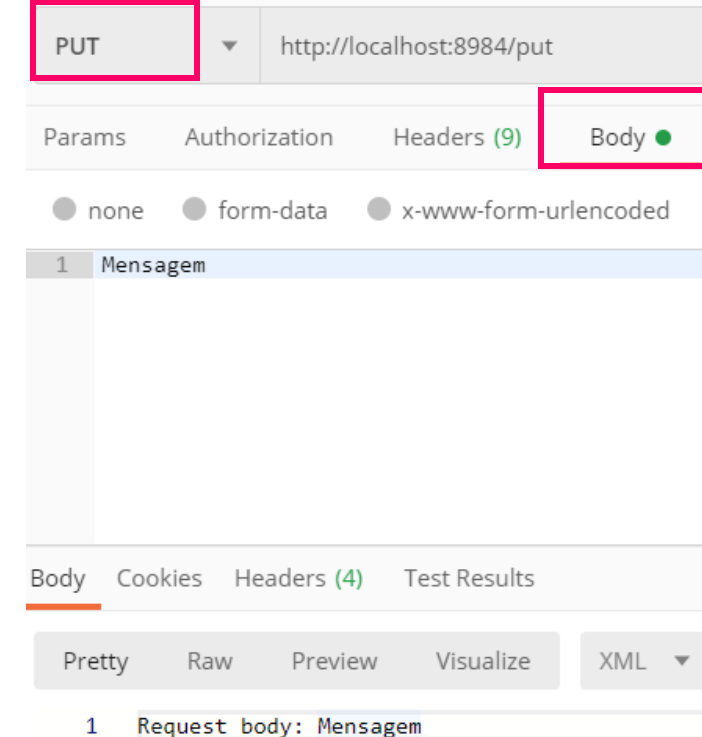
Criar um endpoint com BaseX

Exemplo de invocação utilizando Postman:

Exemplo POST

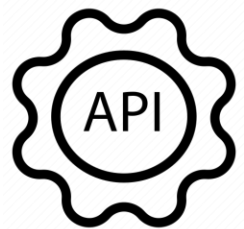


Exemplo PUT



Criar um endpoint com BaseX

Cenário:



Mongo Atlas API (books)

getBook?title=Everyday Italian

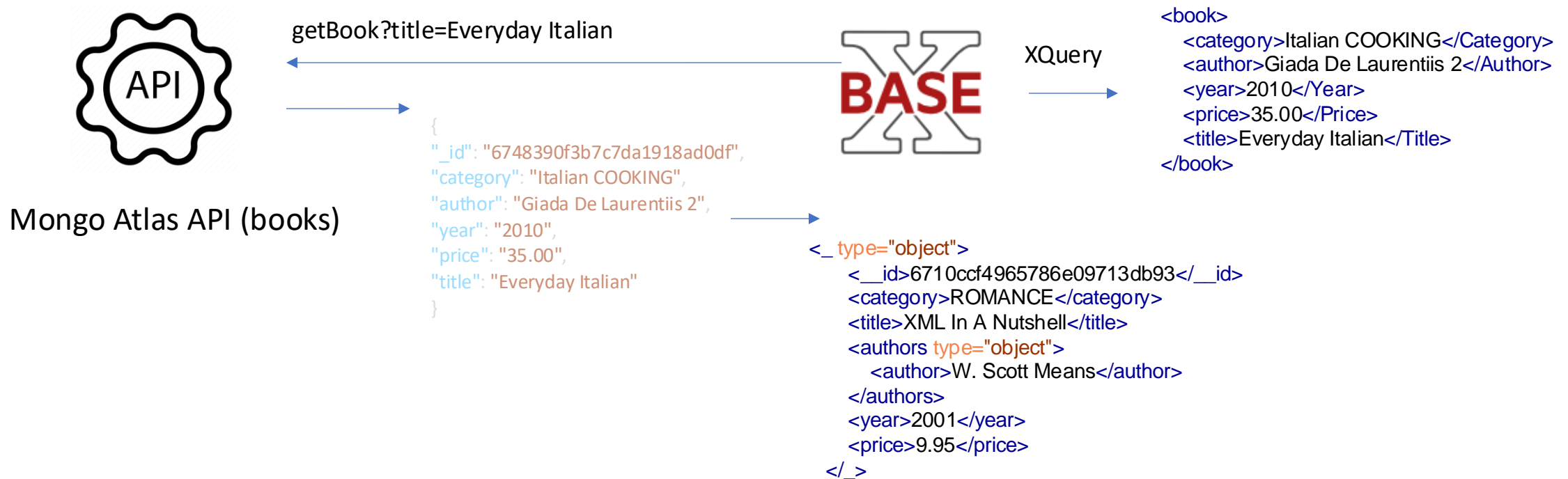
```
{  
  "_id": "6748390f3b7c7da1918ad0df",  
  "category": "Italian COOKING",  
  "author": "Giada De Laurentiis 2",  
  "year": "2010",  
  "price": "35.00",  
  "title": "Everyday Italian"  
}
```

Como converter
automaticamente
para XML?

```
<book>  
  <category>Italian COOKING</Category>  
  <author>Giada De Laurentiis 2</Author>  
  <year>2010</Year>  
  <price>35.00</Price>  
  <title>Everyday Italian</Title>  
</book>
```


Criar um endpoint com BaseX

Cenário:



Criar um endpoint com BaseX

No BaseX, na pasta WebApp, criar um novo ficheiro:books.xqm:

```
module namespace page = 'http://basex.org/examples/web-page';
declare
  %rest:path("getbooks")
  %rest:GET
  function page:getBooks() {
    element bookstore {
      for $book in http:send-request(<http:request method='get'/>, 'https://data.mongodb-api.com/app/data-
docuz/endpoint/getBooks')[2]/json/_
      return element book {
        attribute category { $book/category },
        element title {
          $book/title/text()
        },
        if ($book/authors) then
          element authors {
            for $author in $book/authors/author
            return element author { $author/text() }
          }
        else
          element author { $book/author/text() },
          element year { $book/year/text() },
          element price { $book/price/text() }
      }
    }
  };
```

Criar um endpoint com BaseX

<http://localhost:8080/getbooks>

```
▼<bookstore>
  ▼<book category="ROMANCE">
    <title>XML In A Nutshell</title>
    ▼<authors>
      <author>W. Scott Means</author>
    </authors>
    <year>2001</year>
    <price>9.95</price>
  </book>
  ▼<book category="SCIENCE">
    <title>The Cosmos Explained</title>
    <author>Neil deGrasse Tyson</author>
    <year>2018</year>
    <price>45.00</price>
  </book>
  ▼<book category="WEB">
    <title>XQuery Kick Start</title>
    <author>James McGovern</author>
    <year>2003</year>
    <price>49.99</price>
  </book>
  ▼<book category="COOKING">
    <title>30 Dias para Mudar de Vida, Detox Paleo</title>
    <author>Joana Moura</author>
    <year>2016</year>
    <price>12.79</price>
  </book>
  ▼<book category="WEB">
    <title>Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
```

Referências

Referências Web:

- http://www.swennenhuis.nl/basexfordummies/BaseX_for_dummies.pdf
- https://docs.basex.org/wiki/Main_Page
- <https://docs.basex.org/wiki/RESTXQ>

Livro:

- Lauret A., The Design of Web APIs, Manning, 2019;

HTTP com BaseX