Delphi 10.4.2 Community Edition
Summer Camp 2021

Delphi Works

Introduction to REST Web Services in Delphi

Danny Wind

FireMonkey VCL Android Mac iOS Windows 64-bit RTL Cloud Database

embarcadero

MVP

Delphi®

Delphi
Works

Danny Wind
Delphi MVP, trainer and developer
https://dannywind.nl

Blaise Pascal magazine
Worldwide Delphi and Lazarus magazine
https://blaisepascalmagazine.eu

BLAISE PASCAL MAGAZINE

FireMonkey VCL Android Mac iOS Windows 64-bit RTL Cloud Database

Day 1 – REST Web Services

Delphi Works

Download doc, source and slides:

https://dannywind.nl/get/SummerCamp2021_WebServices1.zip

FireMonkey VCL Android Mac iOS Windows 64-bit RTL Cloud Database

Agenda Summer Camp 2021 – REST Web Services

Each day on at 17:00 - 18:00 CEST (Europe) | 10 AM - 11 AM Central Time (US)

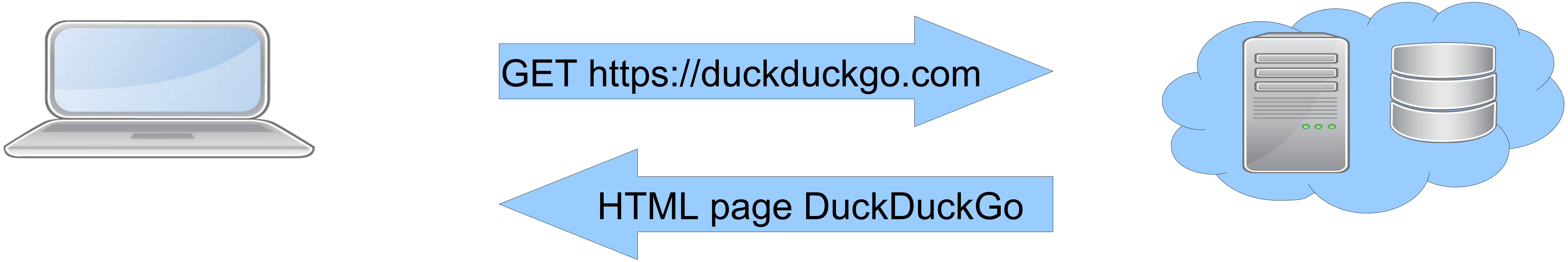
1) Tuesday, Jul 27, 2021
   Introduces some of the concepts you need to know and shows you how to create and consume your own web service in Delphi with just the GET request.

2) Wednesday, Jul 28, 2021
   How to update the data in the web service and how to create in-memory storage for the web service.

3) Thursday, Jul 29, 2021
   How to consume and use your web service from both Delphi clients on Windows and from a web page with JavaScript.

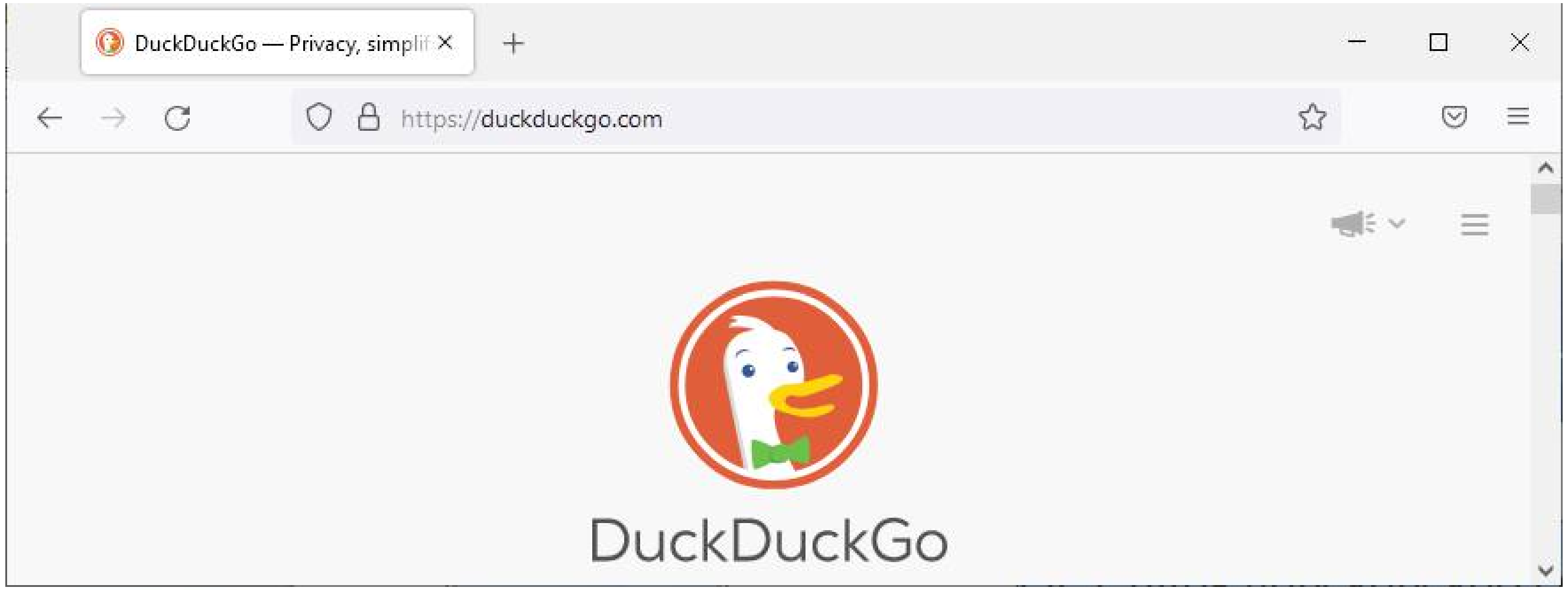
4) Friday, Jul 30, 2021
   Deploying your web service to the Internet Information Services (IIS) server on Windows. (ISAPI)

Browser GET request



GET https://duckduckgo.com

HTML page DuckDuckGo

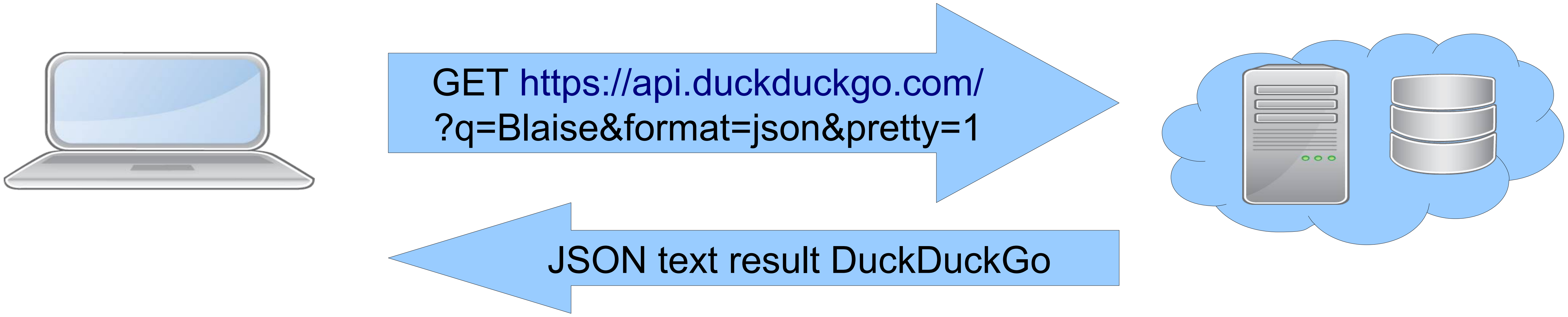


DuckDuckGo — Privacy, simplif

https://duckduckgo.com

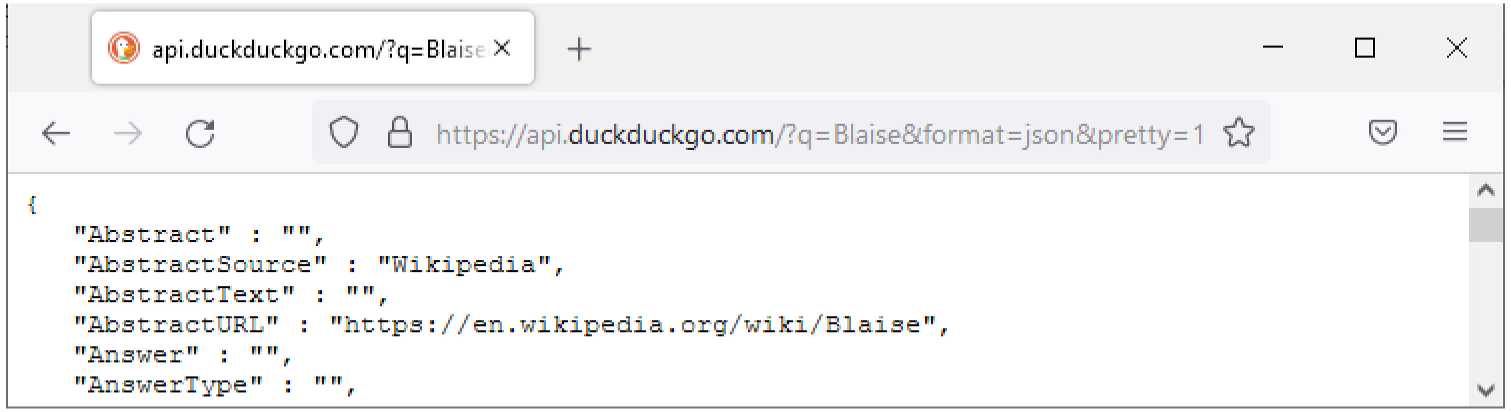
DuckDuckGo

Web Service GET request



GET https://api.duckduckgo.com/?q=Blaise&format=json&pretty=1

JSON text result DuckDuckGo


api.duckduckgo.com/?q=Blaise
https://api.duckduckgo.com/?q=Blaise&format=json&pretty=1

```
{
   "Abstract" : "",
   "AbstractSource" : "Wikipedia",
   "AbstractText" : "",
   "AbstractURL" : "https://en.wikipedia.org/wiki/Blaise",
   "Answer" : "",
   "AnswerType" : "",
```

URL and URI

URL - Uniform Resource Locator

This is the human readable address that a resource (a web service) can be found at.
An example would be https://duckduckgo.com/. Its translated to a physical IP address through DNS. This way the resource can be located over a TCP/IP network.

In analogies an URL would be the home address for the house where your resource lives.

URI - Uniform Resource Identifier

There is also a thing called URI. This identifies a specific resource. If you just go with the idea that this adds a specific resource identifier to retrieve from the URL location, you're not far off.
An example of an URI is https://duckduckgo.com/index.html.

In analogies the URI would be a specific bookcase inside the house.

HTTP – Hyper Text Transfer Protocol

HTTP is the language of the web

HTTP is the protocol used to communicate over TCP/IP with your web service.

In analogies HTTP is a very limited language used to exchange data. It only has nine words.

HTTP request methods (words)

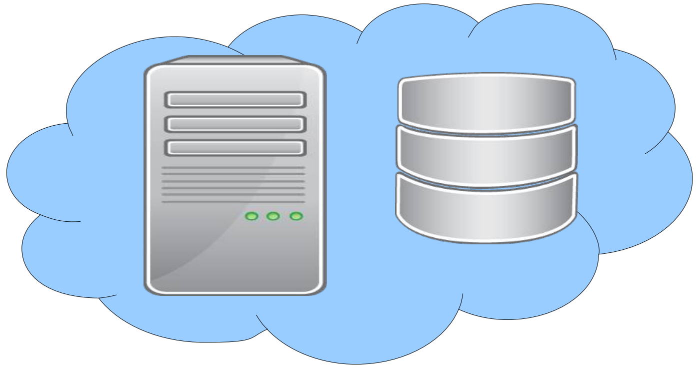
1. GET
2. HEAD
3. POST
4. PUT
5. DELETE
6. CONNECT
7. OPTIONS
8. TRACE
9. PATCH

https://www.w3.org/Protocols/rfc2616/rfc2616.html

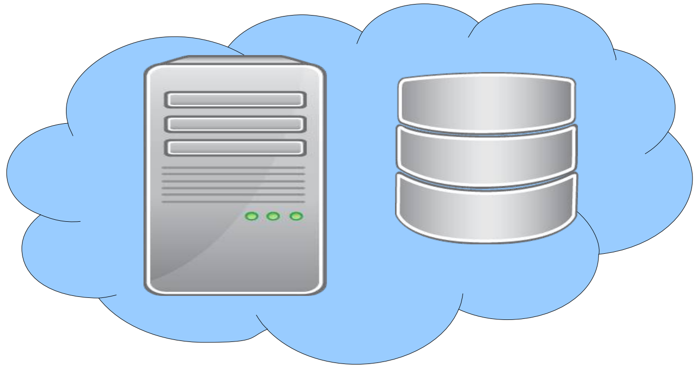
https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol

Why a Web Service?

- Central online storage of data
- Easy recovery and backup for users
- Accessible from every device



Your own Web Service?



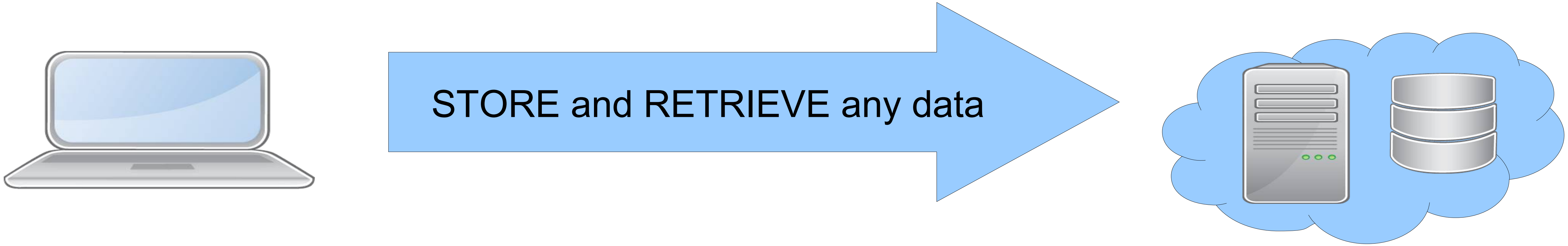
Benefits

- Predictable expenses and low cost
- Full flexibility
- Easy to code and maintain
- Long support window

Downside

- Scalability requires more effort
- Administration

Usage of a Web Service


STORE and RETRIEVE any data

MAP

PDF

BIN

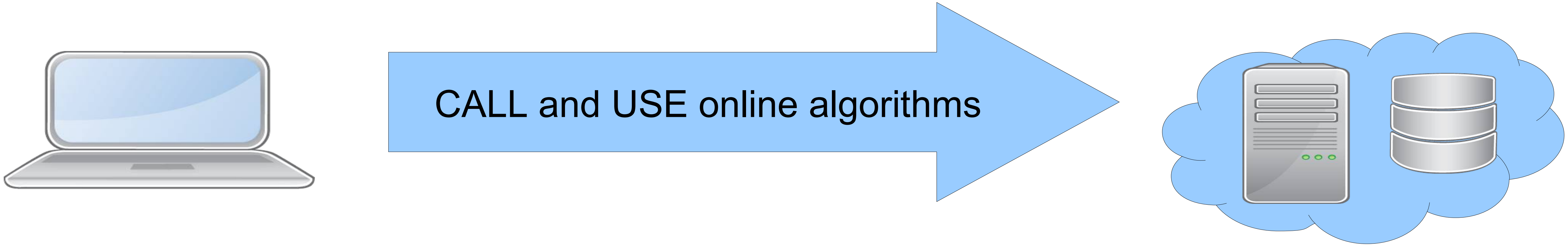
DOC

TXT

JPG

MP4

Usage of a Web Service



CALL and USE online algorithms

Route

Supercomputing

AI

Speech

Image

Video

Calc

Web Service Frameworks

This session

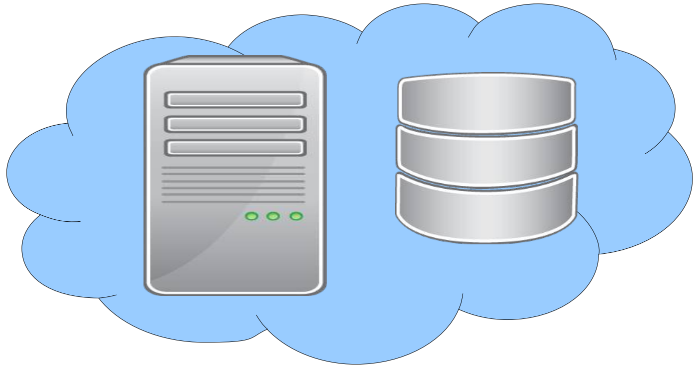
Delphi Web Broker Framework

Other Open Source web service frameworks

MARS Curiosity Framework

mORMot ORM Framework

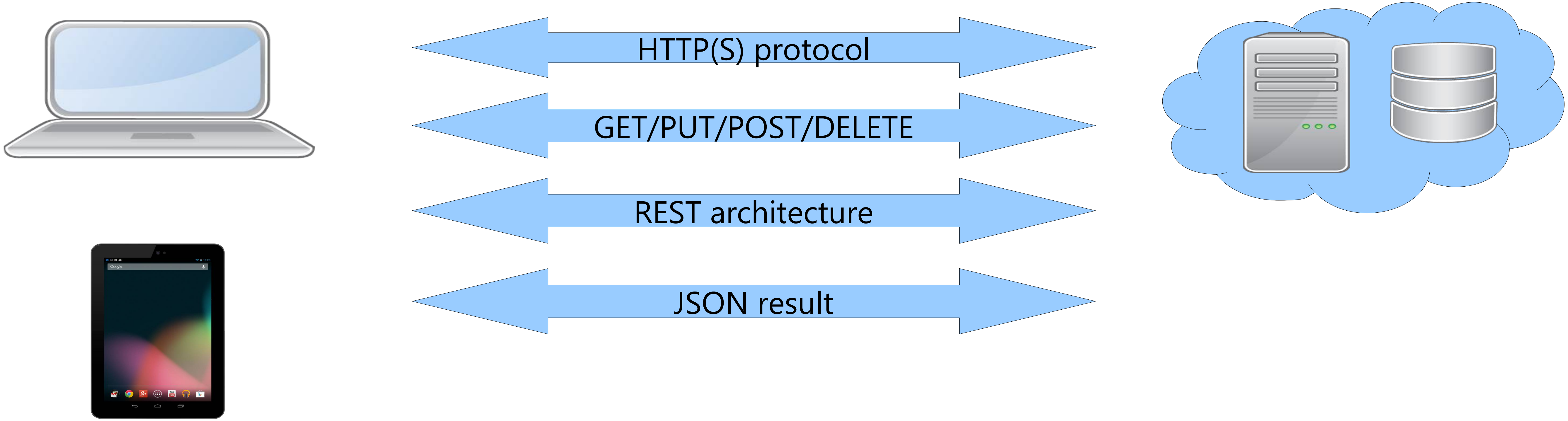
WiRL RESTful library



HTTP Commands in our Web Service

HTTP Command	Usage
HTTP GET idempotent, cacheable	Retrieves data from the resource
usage in our web service	SELECT (get existing record, disallow caching → new data each time)
HTTP POST not idempotent, not cacheable/stale	Appends data to the existing resource
usage in our web service	UPDATE existing (partial update of fields in a record, no update of primary key)
HTTP PUT idempotent, not cacheable/stale	Replace the existing resource or inserts data as a new resource
usage in our web service	INSERT new (or REPLACE) (insert new record with new primary key, or replace)
HTTP DELETE idempotent, not cacheable/stale	Deletes the resource
usage in our web service	DELETE (delete existing record or return error if it doesn’t exist)

REST Web service – protocol stack



HTTP(S) protocol

GET/PUT/POST/DELETE

REST architecture

JSON result

HTTP GET: http://../SomeValue

JSON: {"result":[123]}