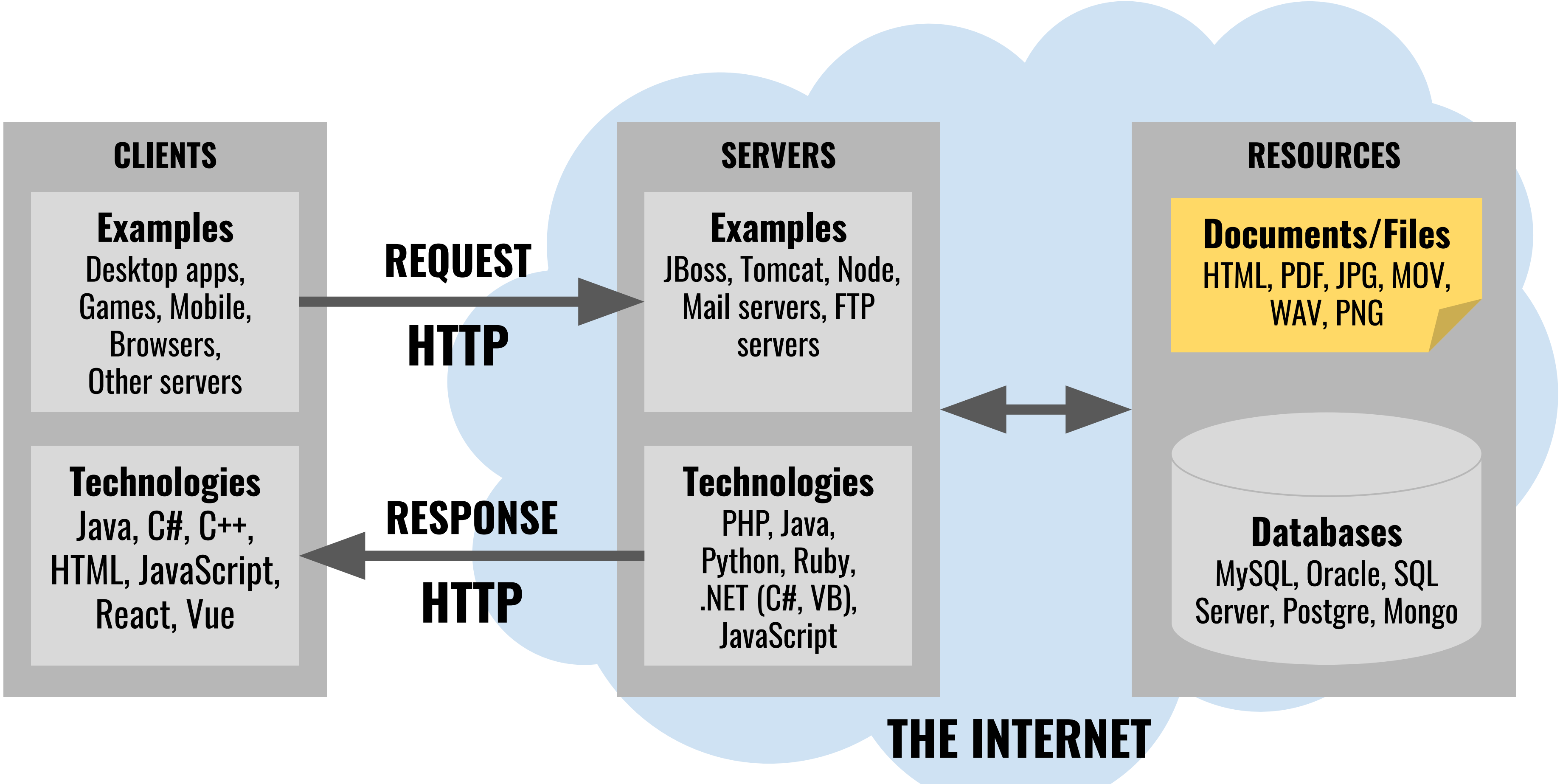
NODE.JS SERVER

The Client Server Architecture



CLIENTS

Examples
Desktop apps, Games, Mobile, Browsers, Other servers

Technologies
Java, C#, C++, HTML, JavaScript, React, Vue

REQUEST

HTTP

RESPONSE

HTTP

SERVERS

Examples
JBoss, Tomcat, Node, Mail servers, FTP servers

Technologies
PHP, Java, Python, Ruby, .NET (C#, VB), JavaScript

RESOURCES

Documents/Files
HTML, PDF, JPG, MOV, WAV, PNG

Databases
MySQL, Oracle, SQL Server, Postgre, Mongo

THE INTERNET

Node.js

- Node.js is a JavaScript runtime used to create middle tier applications using the JavaScript programming language
- Can write JavaScript applications that run on desktop
- Can access file system, databases, networks
- No UI or HTML DOM
- Download and install from https://nodejs.org/en/
- Verify Node.js is installed from the command line

```
$ node
> console.log('hello world')
```

Create a Node.js Application

- **npm** is Node's Package Manager for creating and installing apps
- Choose a directory where to implement a Node.js application, e.g., **my-node-server**

```
$ mkdir my-node-server
$ cd my-node-server
$ npm init
```

- Accept the default configurations

Demo: Create a Hello World Application

- In **hello.js** create a simple **Hello World** JavaScript application

```
console.log('Hello World!');
```

- Run **hello.js** using the **node** JavaScript runtime

```
$ node  hello.js
Hello World!
```

Install Express Library

- ***Express*** is a ***Node.js*** library module for creating middle tier ***HTTP Web*** applications

```
$ npm  install  express  --save
```

Create a Simple Node.js Express Server

- In **server.js**, create an express HTTP server

```
const express = require('express'); // load express
app = express();      // create instance

app.get('/hello',(req, res) =>   // respond HTTP GET
 res.send('hello world'));       // "hello world"

app.listen(3000);          // listen at port 3000
```

GET

Getting a String

```
module.exports = (app) => {

  const getString = (req, res) => {
    res.send("Some String")
  }

  app.get("/get/some/string",
    getString);
}
```

Getting an Object

```
module.exports = (app) => {
  const getString = (req, res) => {
    res.send("Some String")}
  const getObject = (req, res) => {
    const object = {some: "object"};
    res.json(object);
  }
  app.get("/get/some/object", getObject);
  app.get("/get/some/string", getString);
}
```

PATHS

req.params

```
module.exports = (app) => {
  const sayMessage = (req, res) => {
    const msg = req.params["message"];
    res.send(`This is your message:
      ${msg}`);
  }
  app.get("/say/:message",
    sayMessage);
}
```

Can Use Dot Notation Instead

```
module.exports = (app) => {
  const sayMessage = (req, res) => {
    const msg = req.params.message;
    res.send(`This is your message:
      ${message}`);
  }
  app.get("/say/:message",
    sayMessage);
}
```

Parsing Integers

```
const add = (req, res) => {
  const a = parseInt(req.params.a);
  const b = parseInt(req.params.b);
  const c = a + b;
  res.send(`a + b = ${c}`)
}
// http://localhost:3000/add/2/3
app.get("/add/:a/:b", add);
```

QUERIES

Query Parameters

```
const sum = (req, res) => {
 const a = parseInt(req.query.a);
 const b = parseInt(req.query.b);
 const c = a + b;
 res.send(`a + b = ${c}`)
}
// http://localhost:3000/sum?a=2&b=3
app.get("/sum", sum);
```

POST

user-controller.js

```
const register = (req, res) => {
  console.log(req.body)
}
const login = (req, res) => { ... }

app.post("/api/register", register)
app.post("/api/login", login)
```

BODY

Install Body Parser

- Checkout documentation at https://www.npmjs.com/package/body-parser

- Install as follows

```
$ npm install body-parser --save
```

Configure Body Parser

```
var express = require('express')
var bodyParser = require('body-parser')
var app = express()
// parse application/x-www-form-urlencoded
app.use(bodyParser.urlencoded({ extended: false }))
// parse application/json
app.use(bodyParser.json())
```

Alternatively

```
const express = require('express');
const app = express();
app.use(express.json());
```

CORS

Configure CORS

```
app.use(function(req, res, next) {
  res.header("Access-Control-Allow-Origin",
    "https://react-application.herokuapp.com");
  res.header("Access-Control-Allow-Headers",
    "Origin, X-Requested-With, Content-Type, Accept");
  res.header("Access-Control-Allow-Methods",
    "GET, POST, PUT, DELETE, OPTIONS");
  res.header("Access-Control-Allow-Credentials", "true");
  next();
});
```

Alternatively

- Use the cors library

```
$ npm install cors --save
```

```
const express = require('express')
const cors = require('cors')
const app = express()

app.use(cors())
```

SESSION

Express Session

```
$ npm install express-session


const session = require('express-session')
app.set('trust proxy', 1) // trust first proxy
app.use(session({
  secret: 'keyboard cat',
  resave: false,
  saveUninitialized: true,
  cookie: { secure: true }
}))
```

SSL Only in Production

```
const app = express()
const sess = {
  secret: 'keyboard cat',  cookie: {}
}
if (app.get('env') === 'production') {
  app.set('trust proxy', 1) // trust first proxy
  sess.cookie.secure = true // serve secure cookies
}
app.use(session(sess))
```

Configure Session

```
const session = require('express-session')
app.use(session({
  secret: 'keyboard cat',
  // resave: false,
  // saveUninitialized: true,
  // cookie: { secure: true }
  cookie: {}
}))
```

Writing to Session

```
module.exports = (app) => {
  const setAttribute = (req, res) => {
    req.session[req.params.attr] = req.params.value
    res.send(`${req.params.attr} = ${req.params.value}`)
  }
  app.get("/session/set/:attr/:value", setAttribute)
}
```

Reading from Session

```
module.exports = (app) => {
  const getAttribute = (req, res) => {
    const value = req.session[req.params.attr]
    res.send(value)
  }
  app.get("/session/get/:attr", getAttribute)
}
```