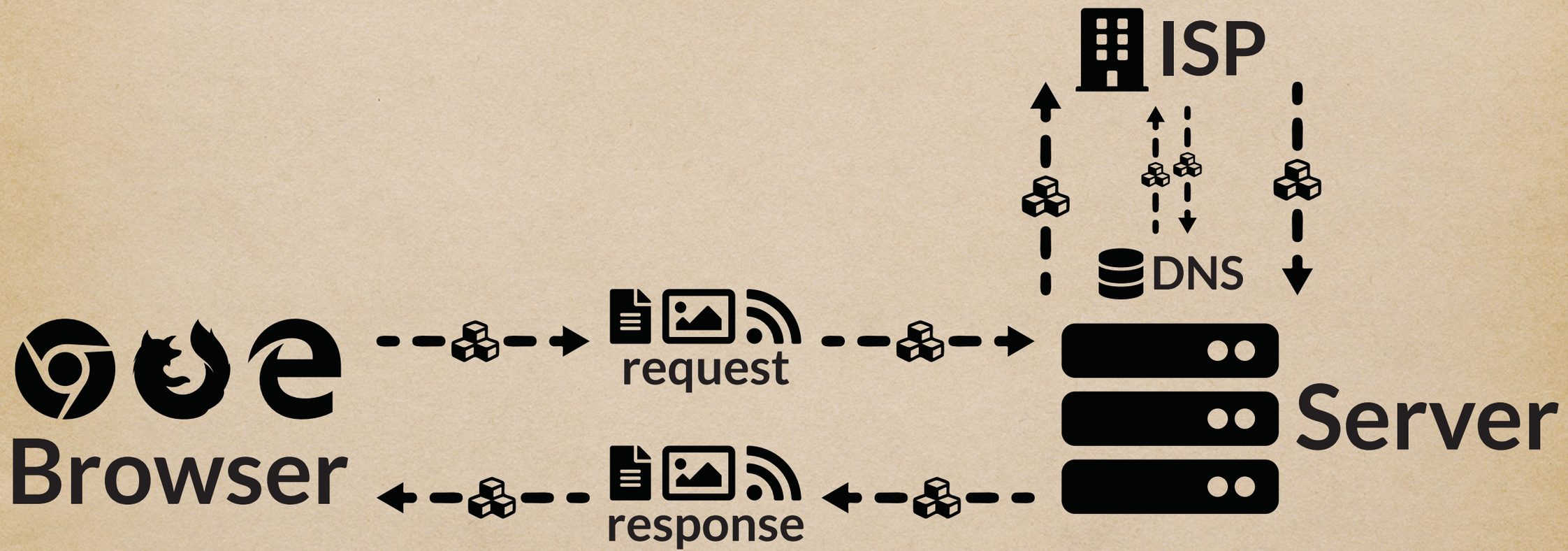
The Internet, MEAN, and NODE.js

A review and an introduction

Review

Review: How the Internet Works

Reflect: How the Internet Works



Reflect: How the Internet Works

```
GET /doc/test.html HTTP/1.1
Host: www.test101.com
Accept: image/gif, image/jpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0
Content-Length: 35

bookId=12345&author=Tan+Ah+Teck
```

Request Line

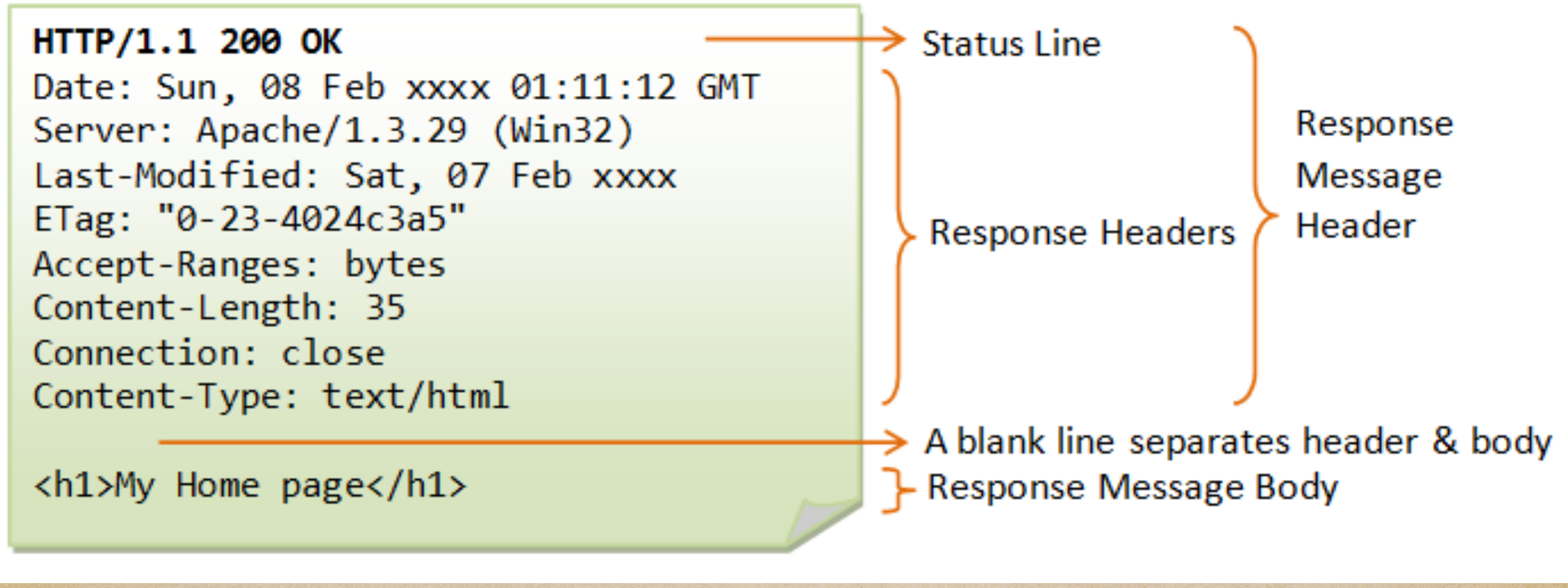
Request Headers

Request Message Header

A blank line separates header & body

Request Message Body

Reflect: How the Internet Works

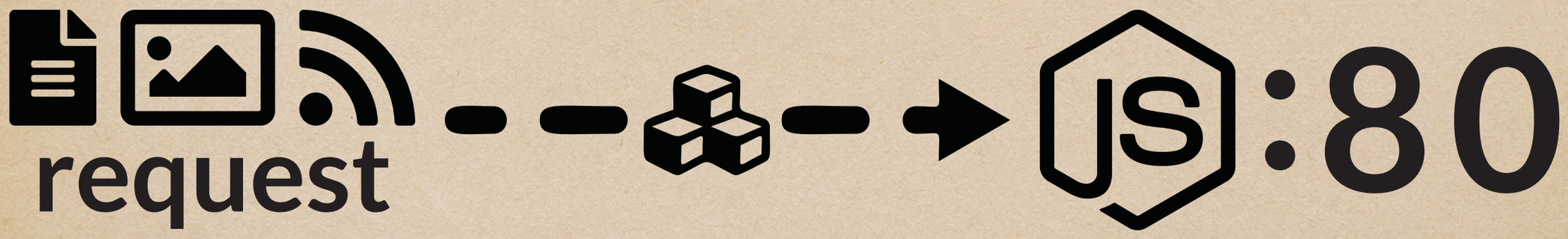


Clients and Servers

- A client is generally stateless
- Servers maintain state
- Clients capability is unreliable
- Servers can be as powerful as you choose
- Clients are an unknown beast
- Servers are controlled environments
- Clients should always be considered insecure
- Servers is responsible for security

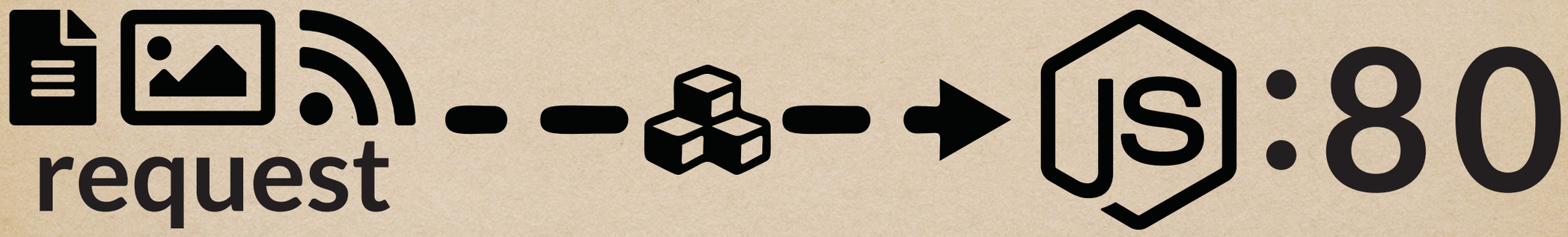
Server Environments

Server Env: Simple



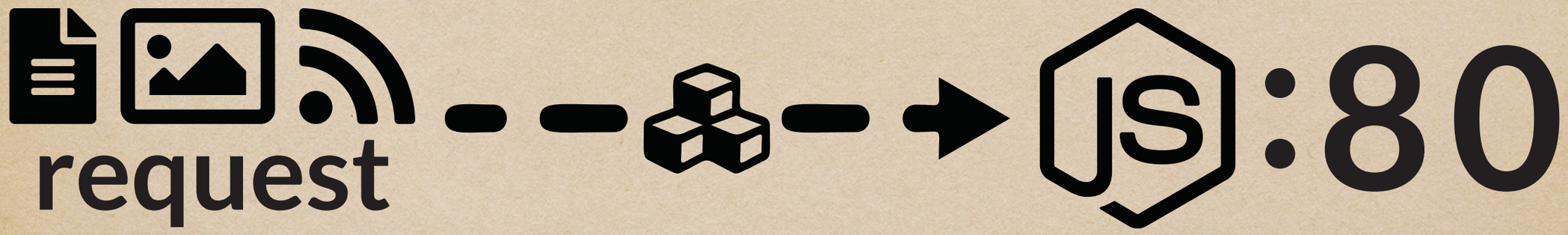
mysite.com/
mysite.com/logo.png
mysite.com/post/1.php

Server Env: RESTful




```
mysite.com/
 /controller
 /method
 /params
```

Server Env: API




```
mysite.com/
/controller
/method
/params
/.json
```

Client Side VS Server Side Programming

Client Side Programming

- Client side programming is programming for the browser
- HTML, CSS, JavaScript, Images, Text, PDFs, anything the browser can render
- Client side programming is not secure or trustworthy

Server Side Programming

- Server side programming is programming for the server

- Node, PHP, Ruby, Python, C++, C#, ASP, JSP, Java

- Server side programming is more secure, but you must NEVER trust information coming from the client - also, Hackers are dicks

ReSTful Architecture

What is ReST

- RESTful (Representational State Transfer)
- Enforces applications to use standard HTTP verbs (GET, PUT, POST, and DELETE)
- GET is strictly for retrieving data and should NEVER be used to permanently modify data
- PUT, POST, and DELETE are used to mutate, create, and destroy data
- Separates data from the presentation of data

Architectural Constraints

- Client-server architecture
- Statelessness
- Cacheability
- Layered system
- Code on demand
- Uniform interface

MVC Design Pattern

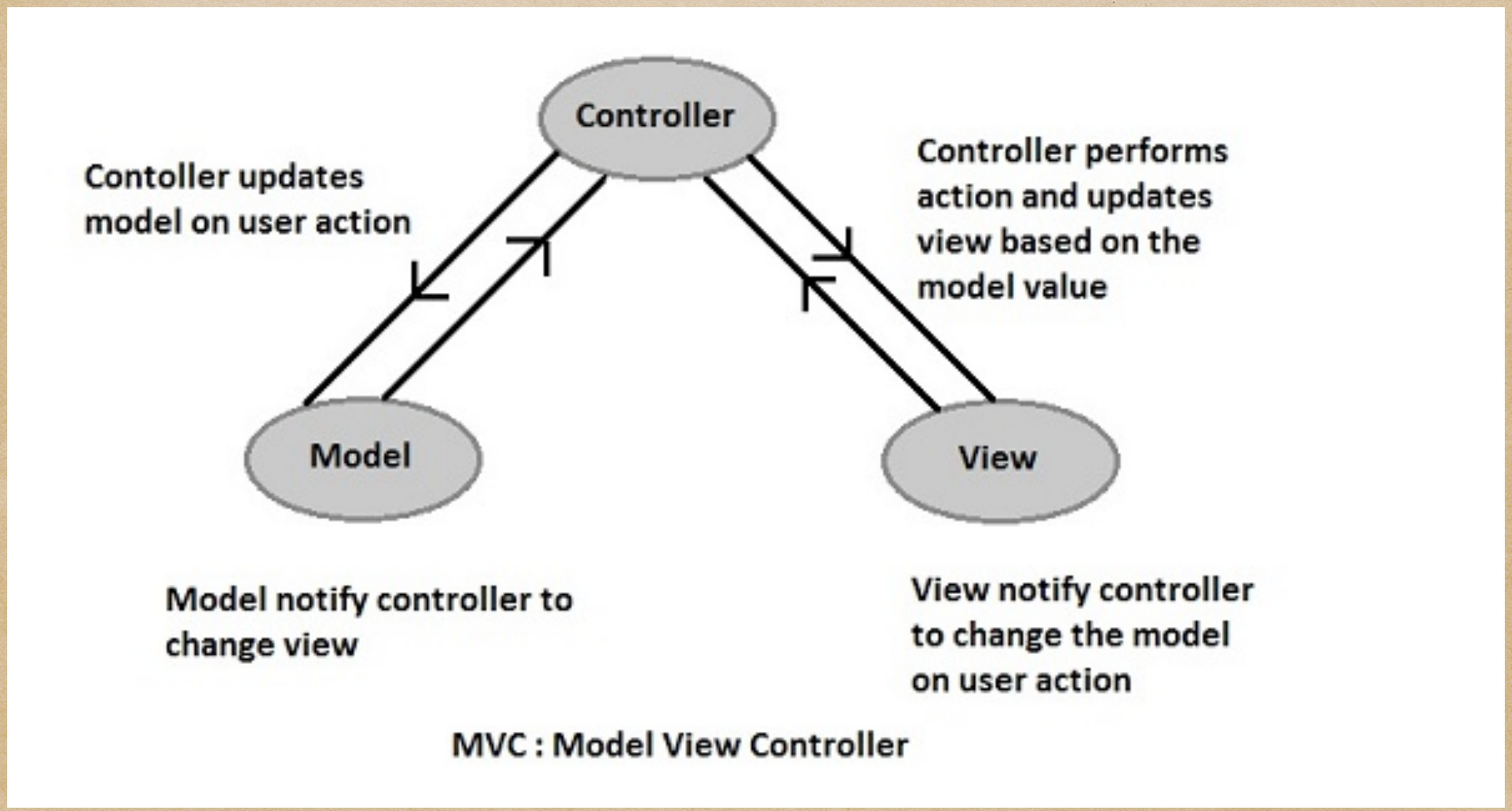
Three Tier Architecture

- Data
- Presentation
- Logic

Three Tier Architecture

- Data (model)
- Presentation (view)
- Logic (controller)

MVC


Controller
Contoller updates model on user action
Controller performs action and updates view based on the model value
Model
View
Model notify controller to change view
View notify controller to change the model on user action
MVC : Model View Controller

MVC Request Flow

1. The URI /controller/method/params
2. The Request (containing the URI) is sent to the Router
3. The Router calls and passes the params to the controller method defined in the URI
4. The response is then returned to the router
5. The router then returns the response to the client

ORMs

ORMs

1. Object Relational Mapper
2. Allows the application to treat data as an object
3. The database table becomes a class known as a Model
4. The class properties (or attributes) are the columns you will find in a table
5. The class methods are common CRUD operations that are performed on a database table

What exactly is a stack?

What is the MEAN stack?

Advantages of MEAN

- A single language is used throughout the application
- All the parts of the application can support and often enforce the use of the MVC architecture
- Serialization and deserialization of data structures is no longer needed because data marshalling is done using JSON objects

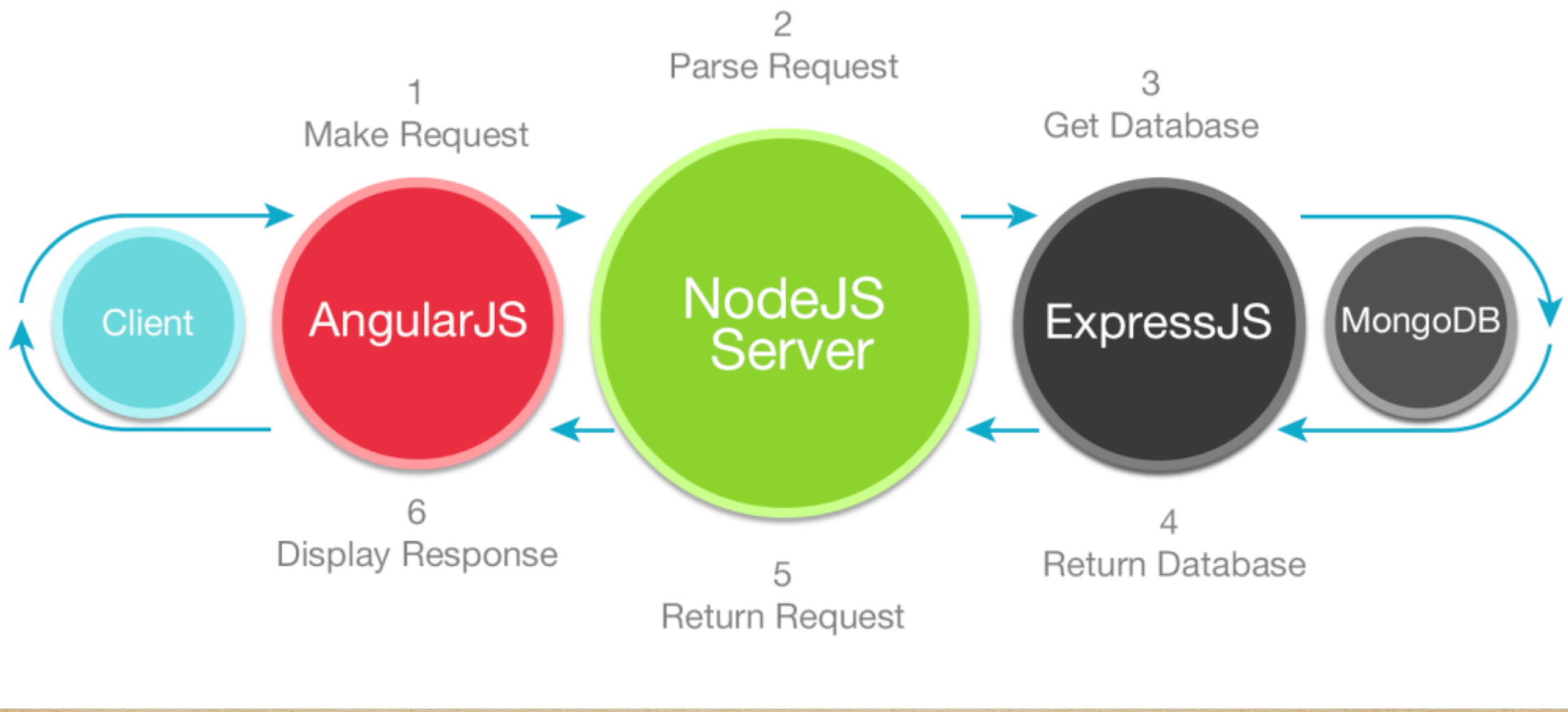
JSON Objects

- JSON stands for JavaScript Object Notation
- JSON represents data in JavaScript arrays
- The modern standard for API development

Example JSON Object

```
[{
  "username":   "youruser1",
  "name":       "Your User 1",
  "phone":      "(555)5557897",
  "email":      "user@example.com",
  "joined":     "2011-04-14 20:55:52",
  "lastactive": "2011-04-15 21:57:21",
  "avatar":     "http://url.to/image.jpg"
},{
  "username":   "youruser2",
  "name":       "Your User 2",
  "phone":      "(123)5557897",
  "email":      "user2@example.com",
  "joined":     "2012-02-14 20:55:52",
  "lastactive": "2012-03-20 21:57:21",
  "avatar":     "http://example.com/user.jpg"
}]
```

How the MEAN stack fits



Let's start with Node :)