RESTful Web API Design with Node.js

Saleh Hamadeh

Video 1.5

Properties of RESTful APIs

In this Video, we are going to take a look at...

- What is REST
- Architectural constraints
  - Client-server
  - Stateless
  - Caching
  - Uniform Interface
  - Layered System
  - Code on Demand

What Is REST?

- REST stands for Representational State Transfer
- REST is an architecture for designing network-based applications
- REST is not a
  - Protocol
  - Framework
  - Standard

Client-server

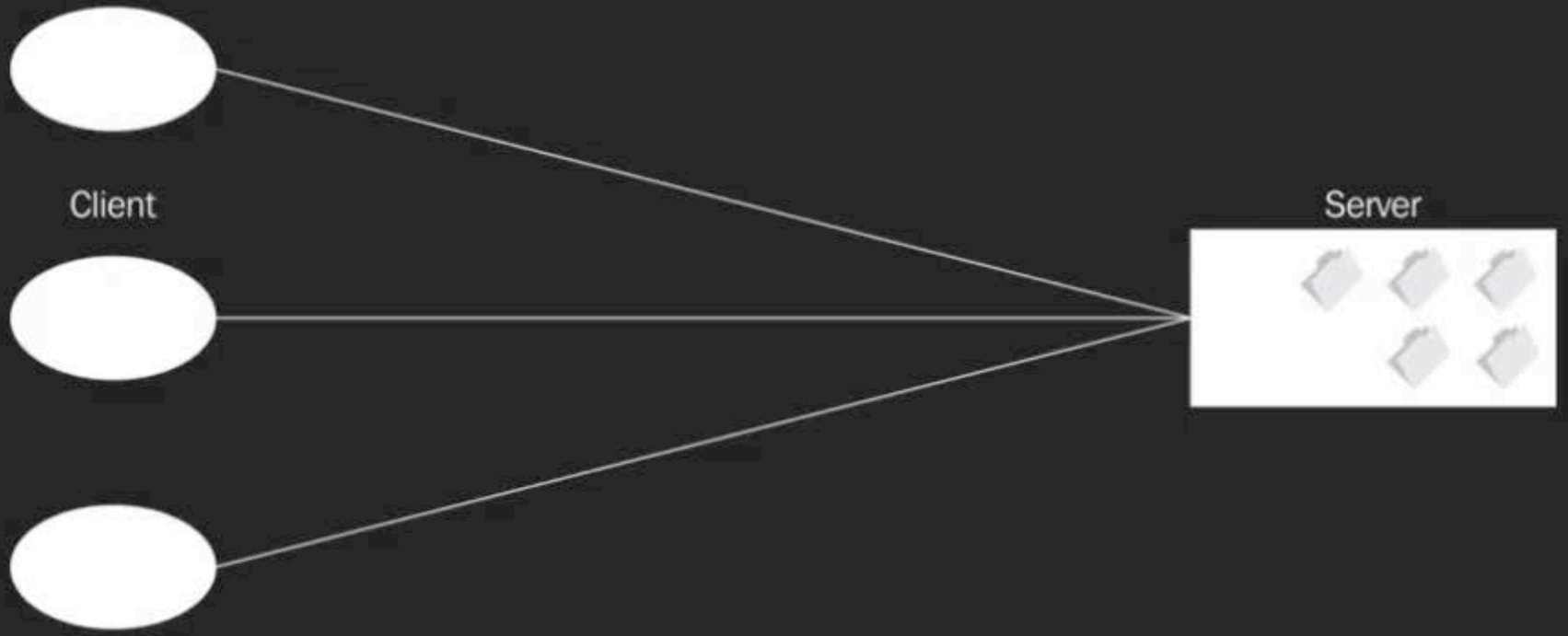
- REST uses a client-server architecture to separate concerns

Benefits of Client-server Architecture

- Portability of client
  - Client handles UI
  - Server handles data storage

Stateless

- REST servers are stateless



Benefits of Stateless Servers

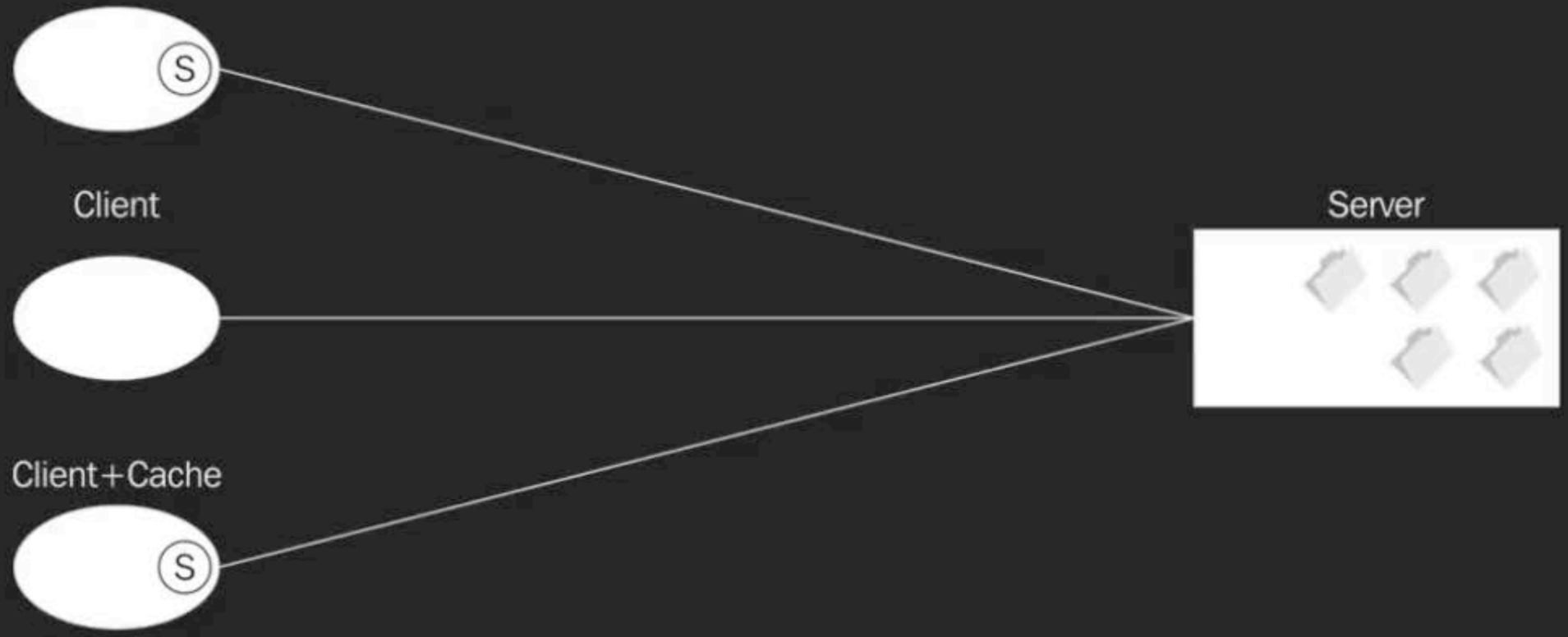
- Visibility
  - Monitoring systems and developers do not need to look beyond the request to trace a bug
- Reliability
  - Easy to recover from system failures
- Scalability
  - Servers can quickly free up resources

Drawbacks of Stateless Servers

- Network Bandwidth
  - Clients send state with every request
- Complexity
  - All clients must handle their states

Caching

- All REST responses must be labeled as cacheable or not



Benefits of Caching

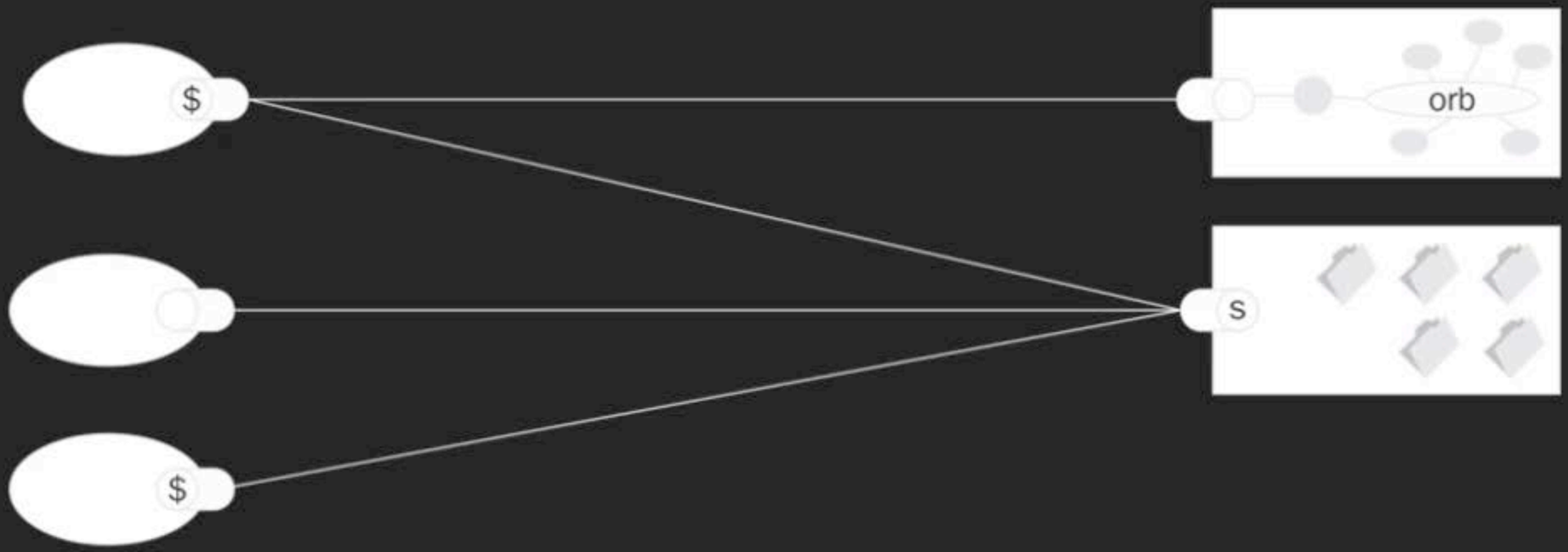
- Performance
  - Stateless + Caching = Caching Anywhere
  - Many requests do not need to go all the way to the server
- Scalability
  - Server gets fewer requests, so it can handle more clients.

Drawbacks of Caching

- Data Reliability
  - Clients may use stale data

Uniform Interface

- Everything is accessed using URL endpoints and data representations



Facets of a Uniform Interface

- Identification of resources
  - All resources are accessed using endpoints on the protocol, such as HTTP
- Manipulation of resources through these representations
  - Resource representations do not need to mimic the data on the server

Facets of a Uniform Interface

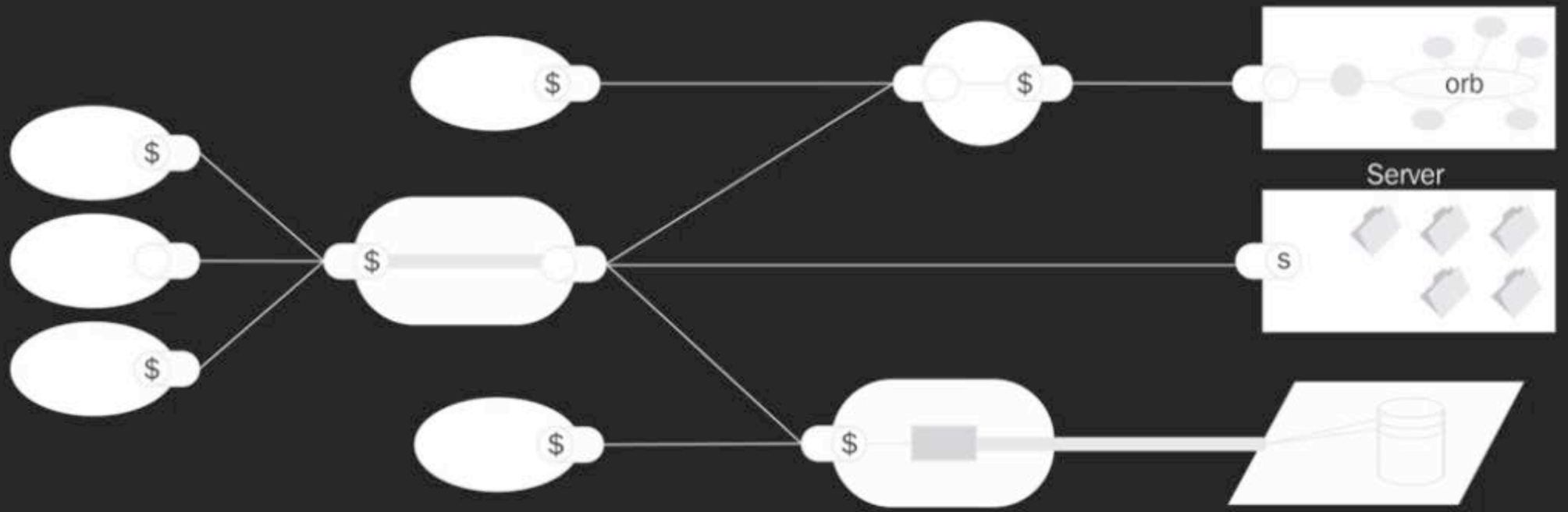
- Self-descriptive messages
  - Server includes metadata, such as Content-Type, to help clients process the responses
- Hypermedia as the engine of application state (HATEOAS)
  - Client only assumes a fixed entry-point to the API, the server tells clients all other available actions through hyperlinks

Drawbacks of a Uniform Inteface

- Degraded Efficiency
  - Some clients may not use all the data that they get in a response

Layered System

- Intermediary components can transform the content of messages



Benefits of a Layered System

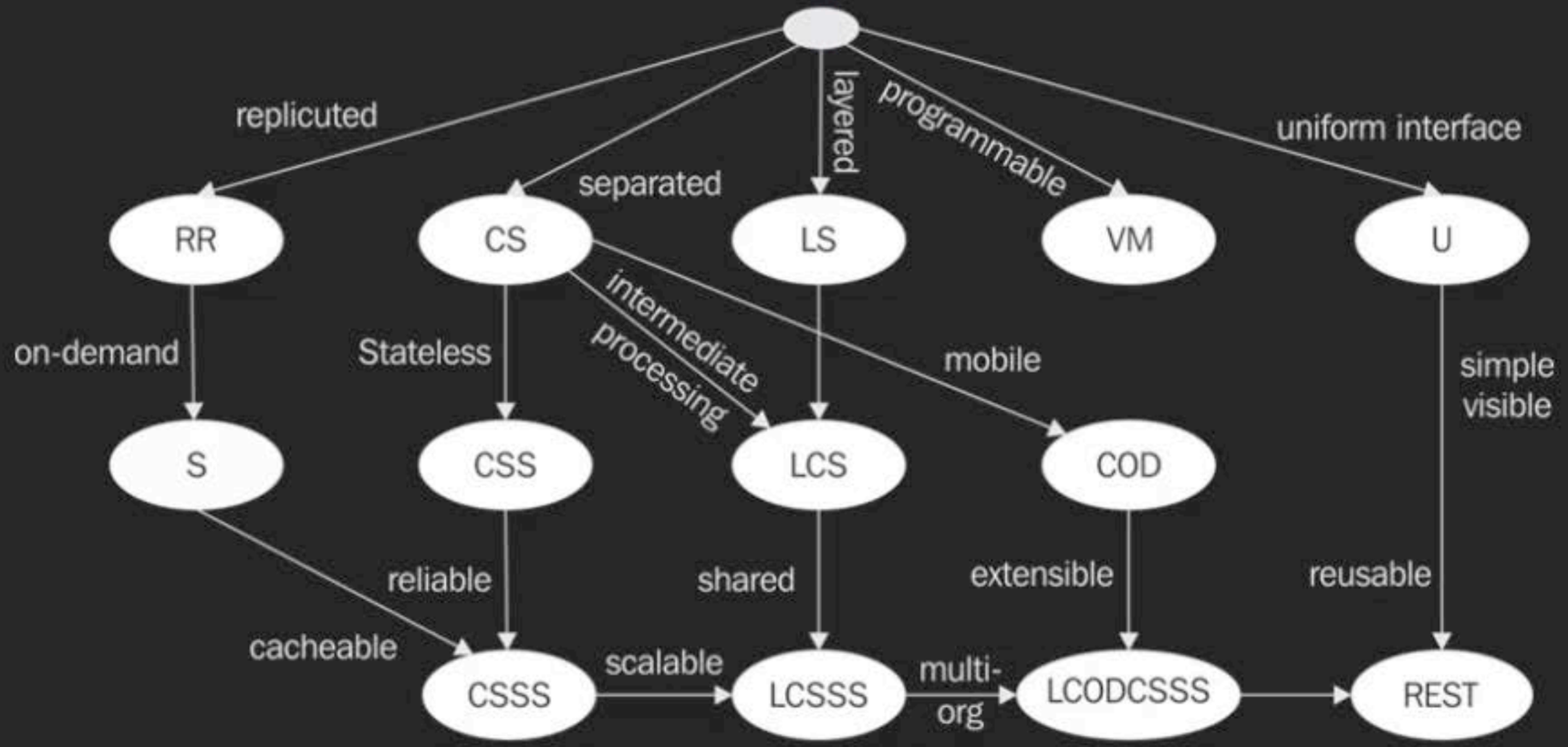
- Encapsulation
  - Layers can be added to simplify an interface to a legacy server
- Scalability
  - Layers enable load balancing
- Security
  - Layers can add access control rules to data crossing a boundary, just like a firewall

Drawbacks of a Layered System

- Latency
  - Adding layers increases latency

Code on Demand (Optional)

- REST clients can be extended by downloading code



Next Video

Setting Up the Environment