

R-Pi

Team Emertxe



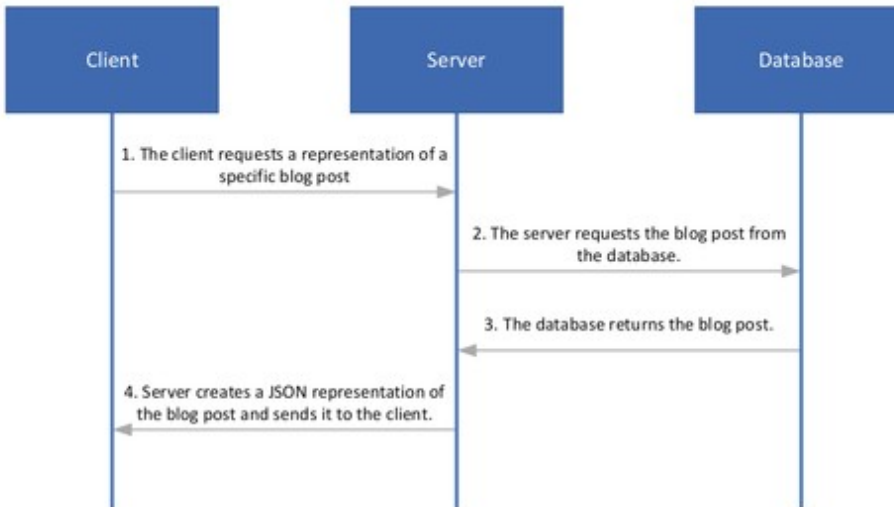
IoT API's
ReST



REST

Introduction

- Acronym: REpresentational State Transfer

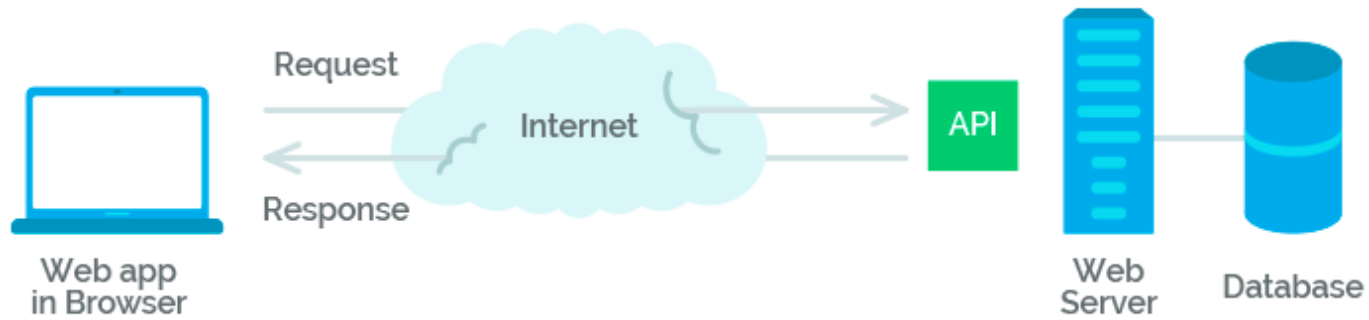


- ???
 - Resources/Nouns
 - Actions
 - Representation
 - State
 - Transfer

REST

Introduction

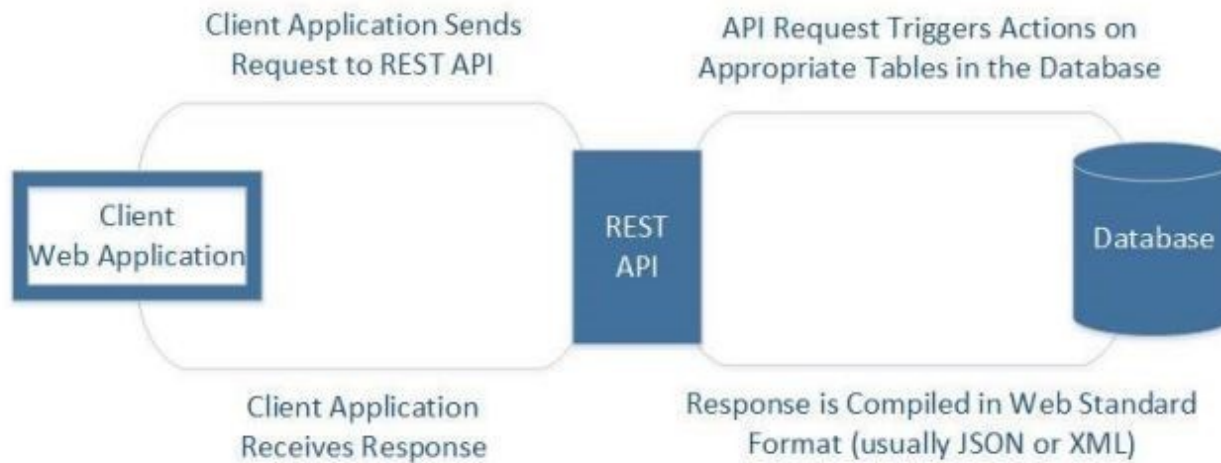
- Acronym: REpresentational State Transfer



REST

Introduction

- Acronym: REpresentational State Transfer

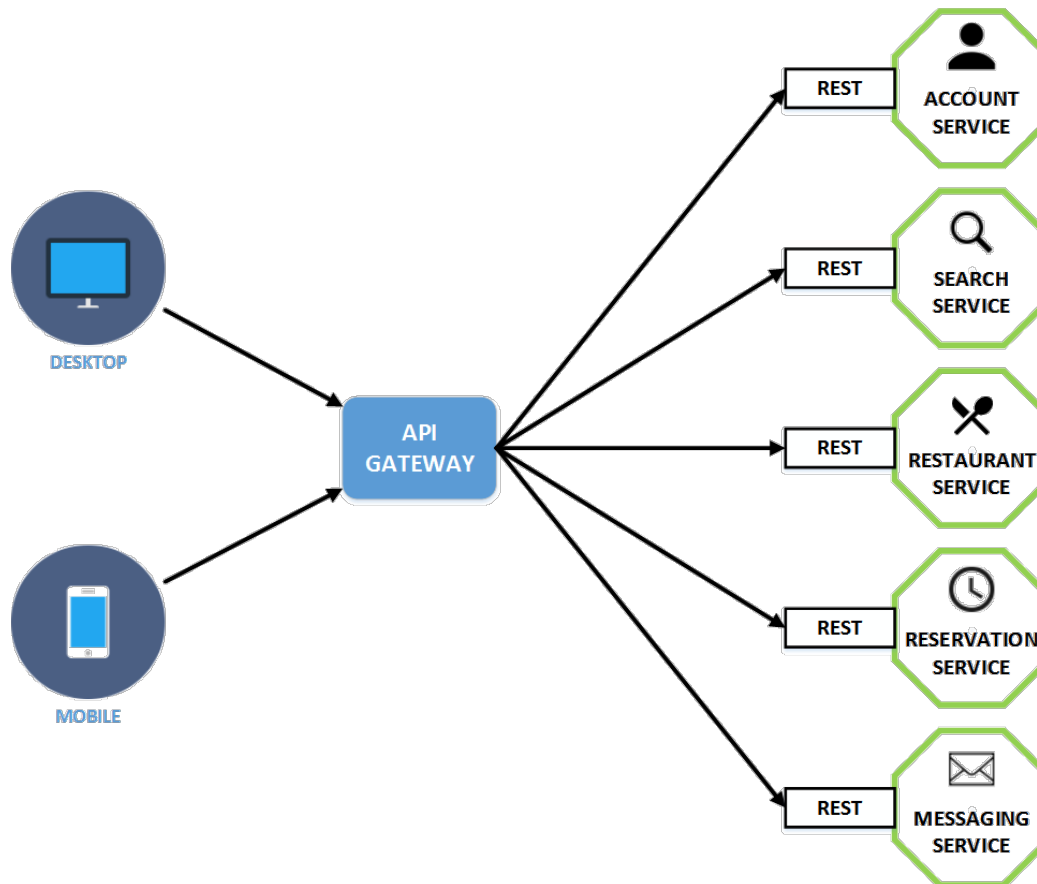


REST is not a standard or protocol, this is an approach to or architectural style for writing API

REST

Introduction

- Acronym: REpresentational State Transfer



REST is not a standard or protocol, this is an approach to or architectural style for writing API

REST

Need

- Separate Client / Server
- Independent of platforms / languages
- Provides flexibility
- It is scalable
- Not constraint to one format
- Built on top of HTTP, can take advantage of HTTP cache
- Easy to use

REST

Understanding



- Is an architectural style
- Restful refers to web services that implement REST
- Simply a program that returns data from the database to the client in a format that it requests
- Based on HTTP, retrieves data from standard HTTP methods,
 - Get
 - Put
 - Post
 - Delete

REST

Constraints

- Client - Server
- Stateless
- Cache
- Uniform Interface
- Layered System
- Code on Demand

REST- Constraints

Client-Server



- REST application should have Client - Server architecture
- Client & Server is decoupled
- Both can evolve independently
- Clients need not worry about the logic / data access layer
- Servers need not know anything about the frontend UI

REST- Constraints

Stateless



- Stateless constraints states that the server does not store any session data
- All the information to understand the request is contained in the request itself
- Improves scalability

REST- Constraints

Cache



- Responses should be cachable, if possible
- It requires that every response should include whether a response can be cachable or not
- For subsequent requests, the client can retrieve from its cache, no need to send the request to the server
- Reduces network latency

REST- Constraints

Uniform Interface



- Key differentiator between REST and NON-REST API's
- 4 elements of UI constraints
 - Identification of Resources
 - Manipulation of Resources through Representations
 - Self-Descriptive Messages
 - Hypermedia as the Engine of Application State
- Promotes generality as all components interacts in the same way

REST- Constraints

Uniform Interface: Identification of Resources



- Identification of Resources
 - Each distinct Web-based concept is known as a resource and may be addressed by a unique identifier, such as a URI.
 - For example, a particular home page URI, like `http://www.emertxe.com`, uniquely identifies the concept of a specific website's root resource.

REST- Constraints

Uniform Interface: Manipulation



- Manipulation of Resources through Representations
 - Clients manipulate representations of resources.
 - The same exact resource can be represented to different clients in different ways.
 - For example, a document might be represented as HTML to a web browser, and as JSON to an automated program.
 - The key idea here is that the representation is a way to interact with the resource but it is not the resource itself.
 - This conceptual distinction allows the resource to be represented in different ways and formats without ever changing its identifier.

REST- Constraints

Uniform Interface: Self-descriptive messages



- Self-descriptive messages
 - › A resource's desired state can be represented within a client's request message.
 - › A resource's current state may be represented within the response message that comes back from a server.
 - › As an example, a wiki page editor client may use a request message to transfer a representation that suggests a page update (new state) for a server-managed web page (resource).
 - › The self-descriptive messages may include metadata to convey additional details regarding the resource state, the representation format and size, and the message itself.
 - › An HTTP message provides headers to organize the various types of metadata into uniform fields.

REST- Constraints

Uniform Interface: Hypermedia



- Hypermedia as the engine of application state (HATEOAS)
 - › A resource's state representation includes links to related resources.
 - › Links are the threads that weave the Web together by allowing users to traverse information and applications in a meaningful and directed manner.
 - › The presence, or absence, of a link on a page is an important part of the resource's current state.

REST- Constraints(Optional)

Code On Demand



- In addition to the data, the servers can provide executable code to the client
- A constraint which enables web servers to temporarily transfer executable programs, such as scripts or plug-ins, to clients.
- The client must be able to understand and execute the code that it downloads on-demand from the server.
- For this reason, code-on-demand is the only constraint of the Web's architectural style that is considered optional.
- Web browserhosted technologies like Java applets, JavaScript, and Flash exemplify the code-ondemand constraint.

THANK YOU