REST architecture (RESTful) web services

Overview

- REST stands for **RE**presentational **S**tate **T**ransfer.
- Introduced by Roy Fielding in 2000.
- Everything is a **resource**.
- Uses the **HTTP** protocol.
- Commonly used to create APIs for web applications.

HTTP request and response

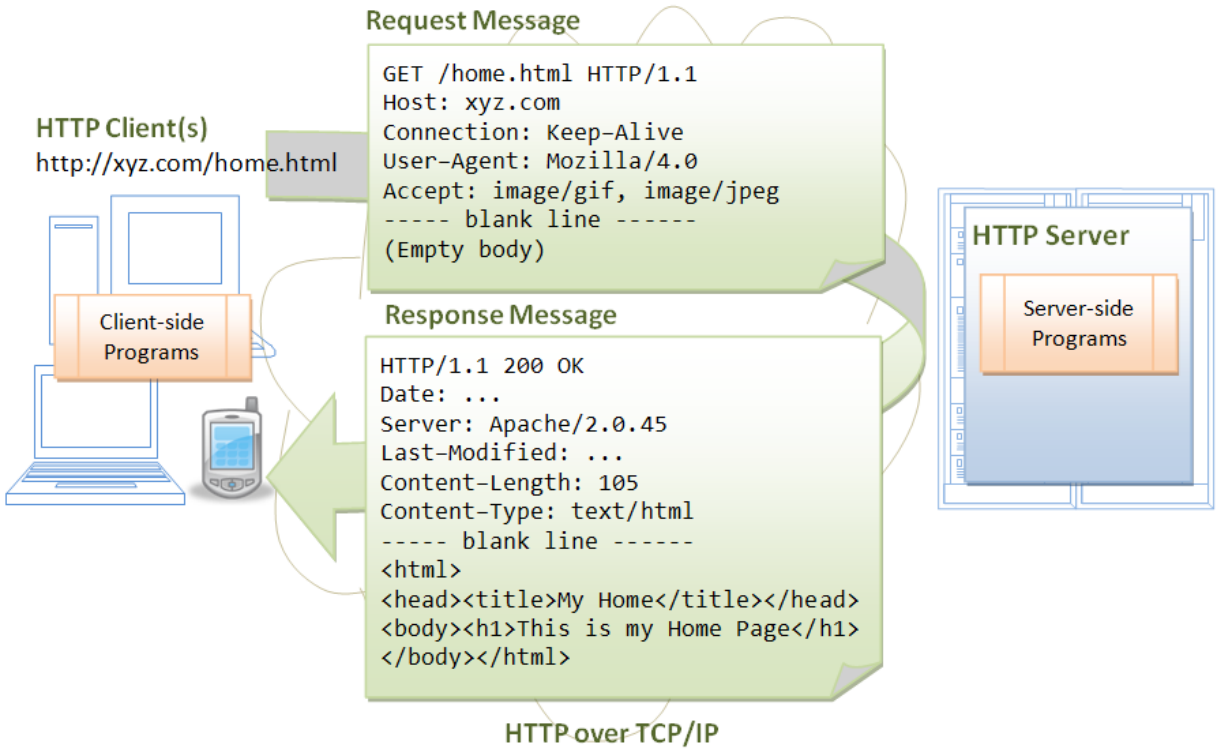
HTTP request

- **Verb**: Indicates the HTTP method such as GET, POST, DELETE, PUT, etc.
- **URI**: Uniform Resource Identifier (URI) to identify the resource on the server
- **HTTP version**: E.g. “HTTP v1.1”.
- **Request header**: Metadata such as client (browser) type, format supported by the client, format of message body, cache settings, etc.
- **Request body**: Message content or resource representation.

HTTP response

- **Status/response code**: Indicates the server status for the requested resource. E.g. 404 means resource not found and 200 means response is ok.
- **HTTP version**: E.g. “HTTP v1.1”.
- **Response header**: Metadata such as content length, content type, response date, server type, etc.
- **Response body**: Response message content or resource representation.

HTTP request and response



Resource

- In REST architecture, everything is a resource.
- A similar concept in OOP would be an entity.
- Possible representations of resources include text, JSON and XML with JSON being the most commonly used format.
- Examples:

XML	JSON

<user>
  <id>1</id>
  <name>Mahesh</name>
  <profession>Teacher</profession>
</user>
	{
	  "id":1,
	  "name":"Mahesh",
	  "profession":"Teacher"
	}

HTTP methods

RESTful web services rely on HTTP methods with the following used most commonly:

- **GET**: Provides a read-only access to a resource.
- **POST**: Used to create a new resources.
- **PUT**: Used to update an existing resource.
- **DELETE**: Used to remove a resource.

HTTP methods

HTTP method	URI	Description
GET	/users	Get all users.
GET	/users/1	Get user with ID 1.
POST	/users	Create a new user.
PUT	/users/1	Update user with ID 1.
DELETE	/users/1	Delete user with ID 1.

REST best practices

- **Use plural noun.** For example, use “users” rather than “user”.
- **Avoid using spaces.** Use hyphen (-) or underscore (_) for long resource names. For example, use “authorized-users” instead of “authorized users”.
- **Use lowercase letters.** Although URI is case-insensitive, it’s a good practice to keep it in lower case letters only.
- **Use HTTP verbs to define the operations.** Do not use the operation name inside the URI.

HTTP method	Bad	Good
GET	http://example.com/api/get-users	http://example.com/api/users
POST	http://example.com/api/create-user	http://example.com/api/users