**1.Difference between HTTP1.1 vs HTTP2:**

**Introduction**

The Hypertext Transfer Protocol, or HTTP, is an application protocol that has been the de facto standard for communication on the World Wide Web since its invention in 1989. From the release of HTTP/1.1 in 1997 until recently, there have been few revisions to the protocol. But in 2015, a reimagined version called HTTP/2 came into use, which offered several methods to decrease latency, especially when dealing with mobile platforms and server-intensive graphics and videos. HTTP/2 has since become increasingly popular, with some estimates suggesting that around a third of all websites in the world support it. In this changing landscape, web developers can benefit from understanding the technical differences between HTTP/1.1 and HTTP/2, allowing them to make informed and efficient decisions about evolving best practices.

After reading this article, you will understand the main differences between HTTP/1.1 and HTTP/2, concentrating on the technical changes HTTP/2 has adopted to achieve a more efficient Web protocol.

**Background**

To contextualize the specific changes that HTTP/2 made to HTTP/1.1, let’s first take a high-level look at the historical development and basic workings of each.

**HTTP/1.1:**

Developed by Timothy Berners-Lee in 1989 as a communication standard for the World Wide Web, HTTP is a top-level application protocol that exchanges information between a client computer and a local or remote web server. In this process, a client sends a text-based request to a server by calling a method like GET or POST. In response, the server sends a resource like an HTML page back to the client.

For example, let’s say you are visiting a website at the domain www.example.com. When you navigate to this URL, the web browser on your computer sends an HTTP request in the form of a text-based message, similar to the one shown here:

GET /index.html HTTP/1.1

Host: www.example.com

This request uses the GET method, which asks for data from the host server listed after Host:. In response to this request, the example.com web server returns an HTML page to the requesting client, in addition to any images, stylesheets, or other resources called for in the HTML. Note that not all of the resources are returned to the client in the first call for data. The requests and responses will go back and forth between the server and client until the web browser has received all the resources necessary to render the contents of the HTML page on your screen.

**HTTP/2:**

HTTP/2 began as the SPDY protocol, developed primarily at Google with the intention of reducing web page load latency by using techniques such as compression, multiplexing, and prioritization. This protocol served as a template for HTTP/2 when the Hypertext Transfer Protocol working group httpbis of the IETF (Internet Engineering Task Force) put the standard together, culminating in the publication of HTTP/2 in May 2015. From the beginning, many browsers supported this standardization effort, including Chrome, Opera, Internet Explorer, and Safari. Due in part to this browser support, there has been a significant adoption rate of the protocol since 2015, with especially high rates among new sites.

From a technical point of view, one of the most significant features that distinguishes HTTP/1.1 and HTTP/2 is the binary framing layer, which can be thought of as a part of the application layer in the internet protocol stack. As opposed to HTTP/1.1, which keeps all requests and responses in plain text format, HTTP/2 uses the binary framing layer to encapsulate all messages in binary format, while still maintaining HTTP semantics, such as verbs, methods, and headers. An application level API would still create messages in the conventional HTTP formats, but the underlying layer would then convert these messages into binary. This ensures that web applications created before HTTP/2 can continue functioning as normal when interacting with the new protocol.

The conversion of messages into binary allows HTTP/2 to try new approaches to data delivery not available in HTTP/1.1, a contrast that is at the root of the practical differences between the two protocols. The next section will take a look at the delivery model of HTTP/1.1, followed by what new models are made possible by HTTP/2.

**HTTP/1.1 — Pipelining and Head-of-Line Blocking:**

The first response that a client receives on an HTTP GET request is often not the fully rendered page. Instead, it contains links to additional resources needed by the requested page. The client discovers that the full rendering of the page requires these additional resources from the server only after it downloads the page. Because of this, the client will have to make additional requests to retrieve these resources. In HTTP/1.0, the client had to break and remake the TCP connection with every new request, a costly affair in terms of both time and resources.

HTTP/1.1 takes care of this problem by introducing persistent connections and pipelining. With persistent connections, HTTP/1.1 assumes that a TCP connection should be kept open unless directly told to close. This allows the client to send multiple requests along the same connection without waiting for a response to each, greatly improving the performance of HTTP/1.1 over HTTP/1.0.

Unfortunately, there is a natural bottleneck to this optimization strategy. Since multiple data packets cannot pass each other when traveling to the same destination, there are situations in which a request at the head of the queue that cannot retrieve its required resource will block all the requests behind it. This is known as head-of-line (HOL) blocking, and is a significant problem with optimizing connection efficiency in HTTP/1.1. Adding separate, parallel TCP connections could alleviate this issue, but there are limits to the number of concurrent TCP connections possible between a client and server, and each new connection requires significant resources.

These problems were at the forefront of the minds of HTTP/2 developers, who proposed to use the aforementioned binary framing layer to fix these issues, a topic you will learn more about in the next section.

**HTTP/2 — Advantages of the Binary Framing Layer:**

In HTTP/2, the binary framing layer encodes requests/responses and cuts them up into smaller packets of information, greatly increasing the flexibility of data transfer.

Let’s take a closer look at how this works. As opposed to HTTP/1.1, which must make use of multiple TCP connections to lessen the effect of HOL blocking, HTTP/2 establishes a single connection object between the two machines. Within this connection there are multiple streams of data. Each stream consists of multiple messages in the familiar request/response format.

1. **Objects and its internal representation in javascript:**

Objects are important data types in javascript. Objects are different than primitive datatypes (i.e. number, string, boolean, etc.). Primitive data types contain one value but Objects can hold many values in form of Key: value pair. These keys can be variables or functions and are called properties and methods, respectively, in the context of an object.

Every object has some property associated with some value. These values can be accessed using these properties associated with them.

var myCar = new Object();

myCar.make = 'Suzuki';

myCar.model = 'Altros';

myCar.year = 1978;

myCar.wheels = 2;

After creating myCar object, the value inside the object can be accessed using keys.

i.e.

myCar.year

Output: 1978

These values can be accessed using brackets notation also.

myCar[year]

Output: 1978

The syntax for adding a property to an object is :

ObjectName.ObjectProperty = propertyValue;

The syntax for deleting a property from an object is:

delete ObjectName.ObjectProperty;

**Object methods:**

An object method is an object property containing a function definition.

i.e.,

Let’s assume to start the car there will be a mechanical functionality.

function(){return ignition.on}

Using the JavaScript Keyword new

The following example also creates a new JavaScript object with four properties:

var person = new Object();

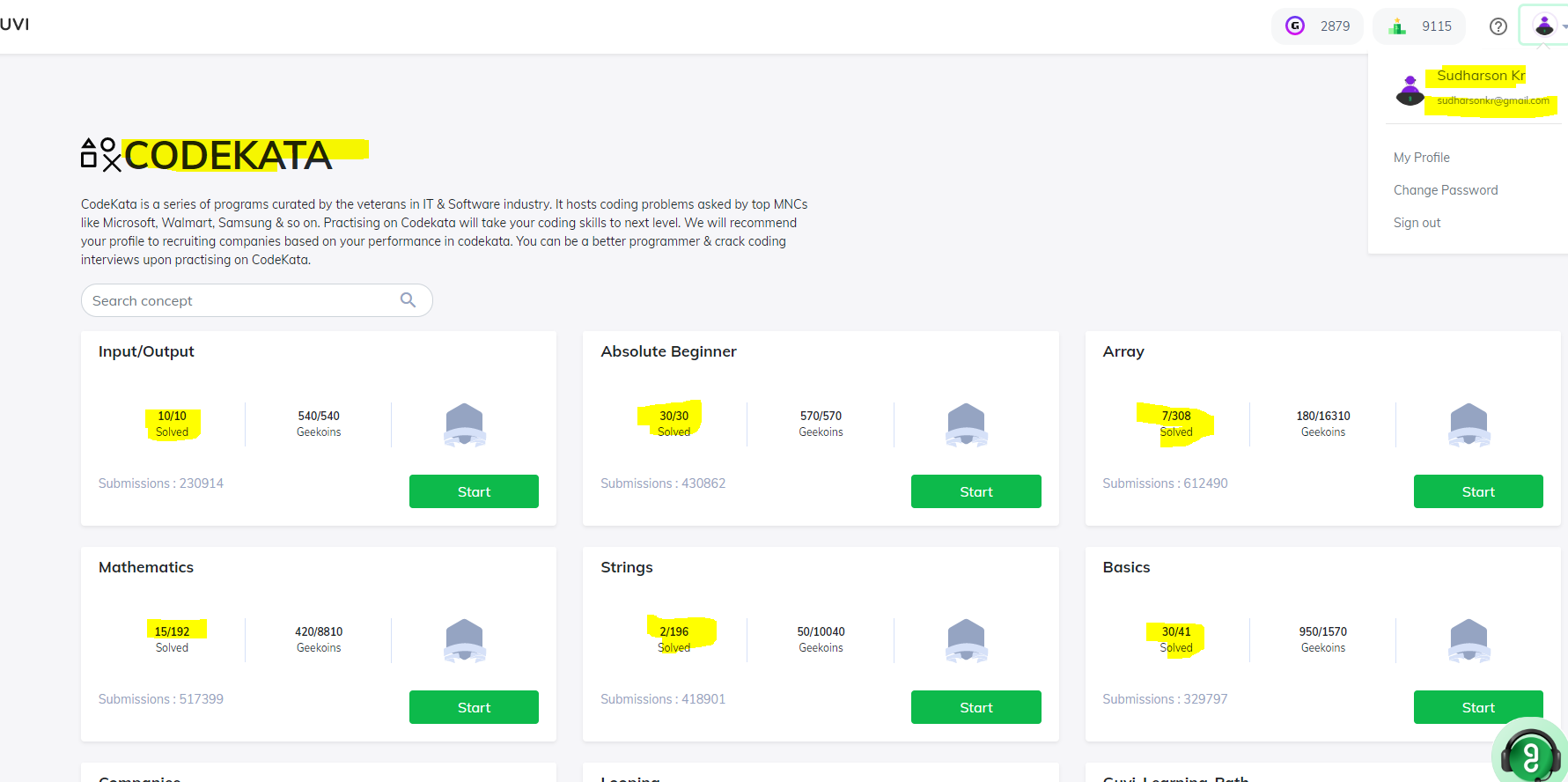
person.firstName = “John”;

person.lastName = “Doe”;

person.age = 50;

person.eyeColor = “blue”;

3.**codekata practice**:



4. **IP address, port, HTTP methods, MAC address**

**IP Address:**

An IP address is a string of numbers separated by periods. IP addresses are expressed as a set of four numbers — an example address might be 192.158.1.38. Each number in the set can range from 0 to 255. So, the full IP addressing range goes from 0.0.0.0 to 255.255.255.255.

IP addresses are not random. They are mathematically produced and allocated by the Internet Assigned Numbers Authority (IANA), a division of the Internet Corporation for Assigned Names and Numbers (ICANN). ICANN is a non-profit organization that was established in the United States in 1998 to help maintain the security of the internet and allow it to be usable by all. Each time anyone registers a domain on the internet, they go through a domain name registrar, who pays a small fee to ICANN to register the domain.

**Types of IP addresses:**

Consumer IP addresses

Private IP addresses

Public IP addresses

Dynamic IP addresses

Static IP addresses

**Two types of website IP addresses**

Shared IP addresses

Dedicated IP addresses

**PORT:**

The need to classify networks has emerged to manage network performance with the rapid increase in internet usage. For all computers to communicate with each other, each computer must have its IP address. These IP addresses have split to form virtual buses. These buses, known as ports, are connections between computer networks. In other words, "ports" are one of the essential factors of data exchange between computers. They are divided into values starting from 0 to 65535 to synchronize many operations at the same time while "ports" are exchanging data. In addition, Ports help network administrators keep track of data and transmit data securely.

As a word meaning Port; is a computer term meaning connection socket, input, socket, used to connect the network to the computer. Ports have a significant role in our daily work on the internet. This role is so important that we can say that internet communication cannot take place without ports. There exists the need to port numbers for network management and network transactions on the internet, such as sending emails, watching videos, surfing the internet.

**HTTP methods:**

The set of common methods for HTTP/1.1 is defined below and this set can be expanded based on requirements. These method names are case sensitive and they must be used in uppercase.

GET

The GET method is used to retrieve information from the given server using a given URI. Requests using GET should only retrieve data and should have no other effect on the data.

HEAD

Same as GET, but transfers the status line and header section only.

POST

A POST request is used to send data to the server, for example, customer information, file upload, etc. using HTML forms.

PUT

Replaces all current representations of the target resource with the uploaded content.

DELETE

Removes all current representations of the target resource given by a URI.

CONNECT

Establishes a tunnel to the server identified by a given URI.

OPTIONS

Describes the communication options for the target resource.

TRACE

Performs a message loop-back test along the path to the target resource.

**MAC address:**

MAC address is the physical address, which uniquely identifies each device on a given network. To make communication between two networked devices, we need two addresses: IP address and MAC address. It is assigned to the NIC (Network Interface card) of each device that can be connected to the internet.

It stands for Media Access Control, and also known as Physical address, hardware address, or BIA (Burned In Address).

It is globally unique; it means two devices cannot have the same MAC address. It is represented in a hexadecimal format on each device, such as 00:0a:95:9d:67:16.

It is 12-digit, and 48 bits long, out of which the first 24 bits are used for OUI(Organization Unique Identifier), and 24 bits are for NIC/vendor-specific.

It works on the data link layer of the OSI model.

It is provided by the device's vendor at the time of manufacturing and embedded in its NIC, which is ideally cannot be changed.

The ARP protocol is used to associate a logical address with a physical or MAC address.

**Types of MAC address:**

Unicast MAC Address

Multicast MAC address

Broadcast MAC address