\* Contructor : hàm khởi tạo ban đầu của mọi đối tượng, trong contructor có property nào thì đối tượng thuộc lớp sẽ có các thuộc tính đó ngay khi khởi tạo.

\* Propertype , Cũng là property của lớp Object, có chức năng thêm được property cho đối tượng được kế thừa từ Object. Được gọi là thuộc tính nguyên mẫu .

Function Person(first , last, age){  
 this.firstName = first;

this.last= last;

this.age= age;

}

Person.nationality = "English"; // khong the them duoc property nationality .

Person.propertype.nationality = "English"; // Person kế thừa từ Object.prototype.

writable : true      // Property value can be changed  
 enumerable : true    // Property can be enumerated (liet ke)  
 configurable : true  // Property can be reconfigured (cau hinh lai)

// Adding or changing an object property  
Object.defineProperty(object, property, descriptor)  
vd : Object.defineProperty(person, "language", {writable:false,value : "NO"});

// Accessing Properties  
Object.getOwnPropertyDescriptor(object, property)  
vd:   
Object.defineProperties(person2, {"Language":{value:"EN"},"Subject":{value:"English"}})

// Returns all properties as an array  
Object.getOwnPropertyNames(object)

Vd:

Object.getOwnPropertyDescriptor(person2,'age')

//return => { value: 222, writable: true, enumerable: true, configurable: true }

// Returns enumerable properties as an array  
Object.keys(object) // return array keys  
  
// Accessing the prototype  
Object.getPrototypeOf(object)  
  
// Prevents(ngan can) adding properties to an object  
Object.preventExtensions(object)  
// Returns true if properties can be added to an object  
Object.isExtensible(object) //check  
  
// Prevents changes of object properties (not values)  
Object.seal(object)  
// Returns true if object is sealed  
Object.isSealed(object)  
  
// Prevents any changes to an object  
Object.freeze(object)  
// Returns true if object is frozen  
Object.isFrozen(object)

\* Property

- Thuộc tính của đối tượng .

- ĐƯợc truy xuất bằng obj.propertyName obj[‘propertyName’]

\* JWT

1. Mục đích sử dụng JWT :

+ Xác thực là một trong những phần quan trọng nhất trong hầu hết các ứng dụng, từ ứng dụng máy tính để bàn đến ứng dụng web hoặc ứng dụng di động.

+ Mục đích chính là xác thực , định danh và bảo mật thông tin .

+ JWT hoạt động tốt trên cả brower và native app.

- Cách JWT hoạt động :

Client -> gửi login data -> server

Server -> tạo JWT -> return JWT -> Client

Client -> gửi Auth JWT cung vs Request header

Server -> Xác nhận JWT -> return Response.

- Server sẽ tạo jwt từ dữ liệu ngừơi dùng với secret key . Trả về cho client JWT đã tạo và client lưu lại, Các request client sẽ gửi kèm JWT vào header. Server sẽ xác nhận JWT vs secret key và trả về response cho client.

- Nơi lưu trữ :

Browser: [Local Storage](https://developer.mozilla.org/en-US/docs/Web/API/Web_Storage_API/Local_storage)

IOS: [Keychain](https://developer.apple.com/documentation/security/keychain_services)

Android: [SharedPreferences](https://developer.android.com/reference/android/content/SharedPreferences)

1. Cấu tạo :

JWT gồm 3 phần : header + payload + signature

+ header :

{

“typ” : “JWT”,

“alg”: “HS256”,

}

- Option header:

- typ : Chỉ ra đây là loại JWT.

- alg : Thuật toán( agorithm), HS256 (HMAC-SHA256) thuật tóán sử dụng secret key.

- cty(Content type) : Nếu sử dụng ký hiệu hoặc mã hóa lồng nhau, đặt thành JWT.

- kid (keyID) : cho biet client su dung key nao de generate token.

- crit : Danh sách các tiêu đề phải được máy chủ hiểu.

- x5u : URL nơi máy chủ có thể truy xuất.

- x5c : Chuỗi chứng chỉ ở định dạng RFC4945 tương ứng với khóa riêng được sử dụng để tạo token signature.

+ payload :

{

"userId": "abcd12345",

"email": ["email@email.com",](mailto:\"contact@bezkoder.com\",)

"iss": "my",

"iat": 1570238918,

"exp": 1570238992

}

- Option payload :

- iss (issuer) : tổ chức phát hành giá trị StringOrURI

- iat (issued at) : time bat dau JWT

- exp (Expiration Time) : thoi gian het han JWT

- sub(Subject) : Xác định chủ thể của JWT.

- aud (Audience) : Xác định những người nhận mà JWT dành cho(client)

- nbf (Not Before) : Xác định thời gian JWT sẽ bắt đầu được chấp nhận để xử lý. Giá trị phải là NumericDate.

- jti (JWT ID) : Định danh duy nhất có phân biệt chữ hoa chữ thường của token .

+ Signature

- Đầu tiên mã hóa(base64) header và payload

- Sau đó hash(băm) ‘Data’ with ‘secret key’.

- Cuối cùng mã hóa(base64) kết quả băm ‘data’ để được ‘signature’

3. Hướng dẫn tạo JWT

- Mã hóa header.

const encodedHeader = base64urlEncode(header);

- Mã hóa payload.

const encodedPayload = base64urlEncode(payload);

- Tạo signature

const data = encodedHeader + '.' + encodedPayload ;

const hashedData = Hash(data, secret);

const signature = Base64UrlEncode(hashedData);

- Result JWT

const token = encodedHeader + "." + encodedPayload + "." + signature;

- Tạo jwt nodejs

https://github.com/norealy/AccessAndRefreshToken/