# JSON Web Token (JWT)

JSON Web Token (JWT) is an open standard (RFC 7519) that defines a compact and self-contained way for securely transmitting information between parties as a JSON object. It is widely used for authentication and information exchange in web applications.

## How to Use JWT?

1. Installation You can install the jsonwebtoken library using npm:

```
npm install jsonwebtoken
```

2. Creating a JWT To create a JWT, you use the jsonwebtoken library's sign method. This method takes a payload (data you want to store in the token), a secret or private key, and optional settings like expiration time.

```
const jwt = require("jsonwebtoken");
const secretKey = "your_secret_key";

const payload = {
  userId: 1,
  username: "exampleuser",
};

const token = jwt.sign(payload, secretKey, { expiresIn: "1h" });
console.log(token);
```

3. Verifying a JWT To verify a JWT received from a client, use the verify method. This method verifies the token's signature and decodes the payload.

```
const token = "your_received_token";

jwt.verify(token, secretKey, (err, decoded) => {
  if (err) {
    console.error("Token verification failed:", err.message);
    return;
  }

  console.log("Decoded token:", decoded);
});
```

4. Decoding a JWT You can also decode a JWT without verifying its signature. This is useful when you want to extract information from the token but don't need to verify its authenticity.

```
const decoded = jwt.decode(token);
console.log("Decoded token:", decoded);
```

## Methods and Options

```
jwt.sign(payload, secretOrPrivateKey, [options, callback])
```

- payload: Data to be included in the token.
- secretOrPrivateKey: Secret key or private key used to sign the token.
- options: Optional parameters such as expiresIn (token expiration time), algorithm (signing algorithm), etc.
- callback: Optional callback function (for asynchronous operation).

```
jwt.verify(token, secretOrPublicKey, [options, callback])
token: Token to be verified.

secretOrPublicKey: Secret key or public key used to verify the token's signature.

options: Optional parameters such as algorithms (list of acceptable algorithms),
issuer, subject, etc.

callback: Callback function that receives the decoded payload or an error.

jwt.decode(token, [options])
token: Token to be decoded.

options: Optional parameters such as complete (boolean, true to return the
complete decoded token).

returns: Decoded payload (not verified for authenticity).
```

---