Ele é semelhante ao While mas a gente precisa ficar criando contador fora do For, já no While sim, o incremento, isso tudo faz parte do For.

for (valor inicial; teste lógico; incremento)

Ele tem esses 3 valores mas nem todas são obrigatórias de você passar então o valor inicial e o contador são opcionais. Eu posso botar meu contador do lado de fora assim como em um while:

```
var numerosAleatorios = [3, 5, 10, 2, 19, 21, 13];

var i = 0
for (; i < numerosAleatorios.length;) {
        console.log(`Sem valor inicial e incremento: ${numerosAleatorios[i]`);
         i++; // Contador
}
```

A gente usa um for pra percorrer um Array ou imprimir números mas nem sempre é preciso ter número envolvido, Array envolvido basta ter uma expressão booleana pra a gente conseguir trabalhar com For.

```
var condicao = true;
        var contador = 1;

        for (; condicao;) {
            console.log(`condição booleana simples: ${contador}`);
            ++contador;
        }
```

Só que deste jeito ele vai entrar em um Loop infinito porque o tempo todo a variável condição é verdadeira então eu preciso definir quando ela vai ser falsa pro programa parar e não ficar em um loop infinito.

```
var condicao = true;
        var contador = 1;

        for (; condicao;) {
            if (contador % 5 === 0) {
                condicao = false;
            }
            console.log(`condição booleana simples: ${contador}`);
            ++contador;
        }
```

Na primeira vez que for divisível por 5 e restar 0 a condicao altera-se para falso, ele vai sair do loop.

No Loop For e no While nós temos o break para usar também, então eu consigo interromper um For.

```
for (var i = 0; i < 10; i++) {
            if (i === 5) {
                break;
            }
            console.log(`Utilizando break ${i}`);
        }
```

Na hora que a variável i chegar no valor 5 ele vai sair do loop. Outra palavra reservada parecida com break é o continue ela também tem no while, se utiliza o continue e o seu loop a partir dele pra baixo não será mais executado, só que ele vai continuar o loop, ou seja, ele vai voltar de novo pro loop e vai interar o próximo valor mas a partir do continue pra baixo ele não executa o código.

```
  for (var i = 0; i < 50; i++) {
            if (i % 2 !== 0) {
                continue;
            }
            console.log('Utilizando continue', i);
        }
```

Se todo número for múltiplo de 2 eu dou um continue, ou seja, a partir dele ele não vai executar o que estiver abaixo que no caso é a mensagem “Utilizando continue’, i. Todo número par para e ele volta pro loop e continua o próximo número, agora se ele não for múltiplo de 2 ele vai imprimir o texto abaixo e vai voltar, então ele imprime somente os números ímpares. Ele é diferente do break, quando você encontra um break além dele não executar o código dali pra baixo ele também segui o loop For, o continue não executa o código do continue pra baixo mas ele volta pro Loop, ou seja, ele pega o próximo número, a próxima condição.

For Of

Ele é um atalho pra quando a gente está interando um Array.

```
var numerosAleatorios = [3, 5, 'Leonardo', 10, 2, 19, 21, 13];
        for (var num of numerosAleatorios) {
            console.log('For of', num);
        }
```

Esse For Of é um atalho para o For comum, porque ele vai percorrer o Array e já vai armazenar na váriavel que a gente coloca o nome (no nosso num) o valor da posição que ele está percorrendo, então o 3 está na 1° posição então ele já armazena o valor da posição na variável (num).

Com o For normal seria assim:

```
for (var i = 0; i < numerosAleatorios.length; i++) {
            console.log('For normal', numerosAleatorios[i]);
        }
```

Em vez de você ter que especificar o Array e dizer a posição que você quer imprimir o For Of faz isso automaticamente, ele automaticamente armazena o valor da posição da posição do Array dentro da sua variável.

FOR DENTRO DE OUTRO FOR

```
for (var i = 1; i <= 5; i++) {
            for (var j = 1; j <= 50; j++) {
                  console.log(`i: ${i}; j: ${j}`);
            }
}
```

Aqui ele imprime a posição da J e da I. Ele vai entrar no 1° For i vai valer 1 e vai seguir, quando ele chegar no 2° For é um loop de 1 até 5 então ele vai executar 5 vezes enquanto o i é = 1 o J vai valer 1, 2,3,4, 5 e aí terminou volta pro 1° For aí o i vai valer 2 e aí entra no 2° For e aí enquanto ele vai valer 2 o J vai ser 1, 2, 3, 4, 5 novamente. Então o 1° For vai executar 1 vez e o 2° vai executar quantas vezes a gente definiu e assim vai, executa 1 vez, executa todas as vezes que eu defini (2° For), sai, executa mais uma vez (1° For), entra no 2° For executa quantas vezes eu defini e assim vai.

```
   for (var i = 1; i <= 5; i++) {
            for (var j = 1; j <= 50; j++) {
                if (j % 2 === 0) {
```

```
                continue;
            }

            if (j > 7) {
                break;
            }
            console.log(`i: ${i}; j: ${j}`);
        }
    }
```