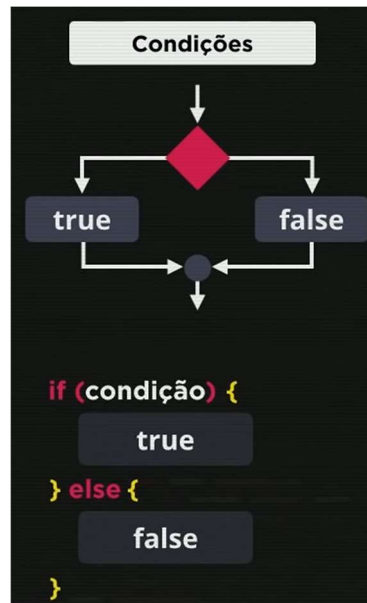


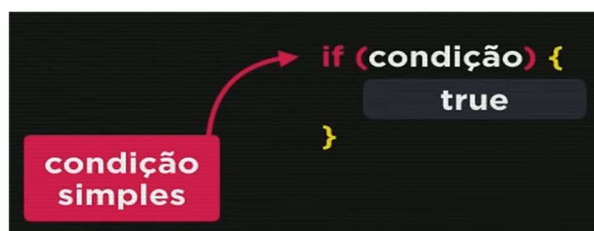
Estruturas condicionais

As estruturas condicionais estão ligadas à tomada de decisão de um algoritmo. Ao utilizar expressões que retornam **verdadeiro** ou **falso**, o algoritmo executa o bloco de comandos relativos a este resultado.



Condição é uma estrutura de controle muito importante para a programação, existem alguns tipos de condição, e vamos ver as diferenças entre eles.

Vamos ver uma condição simples, que ela só tem um tipo de bloco, apenas o bloco do verdadeiro. O bloco do falso não existe.



E essa estrutura, por ser simples, é chamada tecnicamente, de condição simples, se a condição for falsa, nada vai acontecer em especial, o fluxo do programa continua funcionando sem problema.

```
if (condição) {  
    true  
} else {  
    false  
}
```

Mas nesta outra condição, caso seja verdadeira, alguma coisa será feita, e caso seja falsa, outra coisa será feita.

Vamos ver como representar esses dois tipos de condição no Visual Studio Code.

Vamos instalar uma extensão chamada node.exec, ela executa o Node.js quando pressionamos a tecla F8 automaticamente.

Vamos criar uma nova pasta e nomear de condicoes, vamos criar um arquivo chamado exercicio.js, que é de JavaScript.

Vamos testar a extensão, vamos escrever na tela, e para escrever na tela no Node.js não funciona o **document.write**, temos que fazer **console.log** para ele poder escrever no console.

Para testar vamos escrever **o console funcionou corretamente**.

```
console.log('O console funcionou corretamente!')
```

Para executar o código JavaScript, basta apertar a tecla F8, e ele mostra no console a mensagem.

Agora vamos criar uma variável de velocidade, vamos chamar de **vel**, que vai ser a velocidade de um carro, e supor que o carro está andando a 60.5 km por hora.

```
var vel = 60.5
```

Agora vamos utilizar essa variável, quero interpolar com a frase, **a velocidade do seu carro é**.

```
console.log(`A velocidade do seu carro é ${vel}km/h`)
```

Vamos executar, pressione F8.

```
A velocidade do seu carro é 60.5km/h
```

Vamos colocar outro console console.log, **dirija sempre usando cinto de segurança**.

```
console.log('Dirija sempre usando cinto de segurança')
```

Agora, com esses três comandos, temos comandos sequenciais, não existe a possibilidade de executar só um comando, a não ser que coloque um comentário.

```
A velocidade do seu carro é 60.5km/h  
Dirija sempre usando cinto de segurança
```

Se quiser colocar outro `console.log` para multar assim, **você ultrapassou a velocidade permitida, multado!**.

```
console.log('Você ultrapassou a velocidade permitida, MULTADO!')
```

Ao executar esse comando, vamos ser obrigados a sempre multar qualquer pessoa.

Ao executar, temos todos os códigos rodando pois são sequenciais e com isso apresenta também que foi MULTADO!.

```
A velocidade do seu carro é 60.5km/h  
Você ultrapassou a velocidade permitida, MULTADO!  
Dirija sempre usando cinto de segurança
```

Supondo agora que uma pessoa passe a 20km por hora velocidade essa que não tem como multar ninguém, mas se eu executar o código desta maneira, ele sempre vai dizer, você ultrapassou a velocidade permitida.

```
A velocidade do seu carro é 20km/h  
Você ultrapassou a velocidade permitida, MULTADO!  
Dirija sempre usando cinto de segurança
```

O que vamos fazer é criar uma condição, **if**, se, vamos supor que o limite de velocidade dessa estrada seja 60 km por hora, for maior, maior do que 60, ele será MULTADO!

```
if(vel >60){  
    console.log('Você ultrapassou a velocidade permitida, MULTADO!')  
}
```

Estamos automatizando a programação, esse você ultrapassou a velocidade permitida, você está multado, só vai acontecer se a velocidade for maior do que 60.

Esse comando que está dentro das chaves, só vai acontecer se a velocidade for acima de 60.

Por exemplo, agora a velocidade é 60.

```
A velocidade do seu carro é 60km/h
Dirija sempre usando cinto de segurança
```

Agora a velocidade é 80.

```
A velocidade do seu carro é 80km/h
Você ultrapassou a velocidade permitida, MULTADO!
Dirija sempre usando cinto de segurança
```

Esse comando só vai acontecer se for verdade. Essa é uma condição simples, porque ela não tem **senão, else**.

Agora vamos fazer uma condição composta. Vamos começar um código novo, criar uma variável país recebe Estados Unidos, e verificar se o país for igual a Brasil, escreve na tela brasileiro, se não estrangeiro, isso é, se o país da pessoa for Brasil, ela é brasileira, se não é estrangeiro

```
ar país = 'EUA'
console.log(`Vivendo em ${país}`)
if (país == 'Brasil') {
  console.log('Você é Brasileiro!')
}
else {
  console.log('Você é Estrangeiro')
}
```

Vamos executar o programa, o país é Estados Unidos, você é estrangeiro.

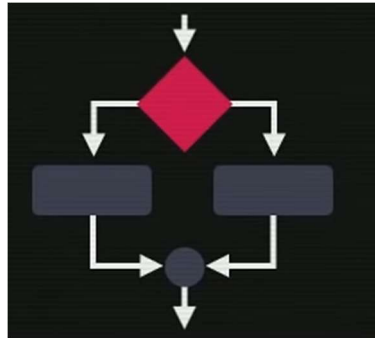
```
Vivendo em EUA
Você é Estrangeiro
```

Se o país for Brasil, vivendo em Brasil, você é brasileiro.

```
Vivendo em Brasil
Você é Brasileiro!
```

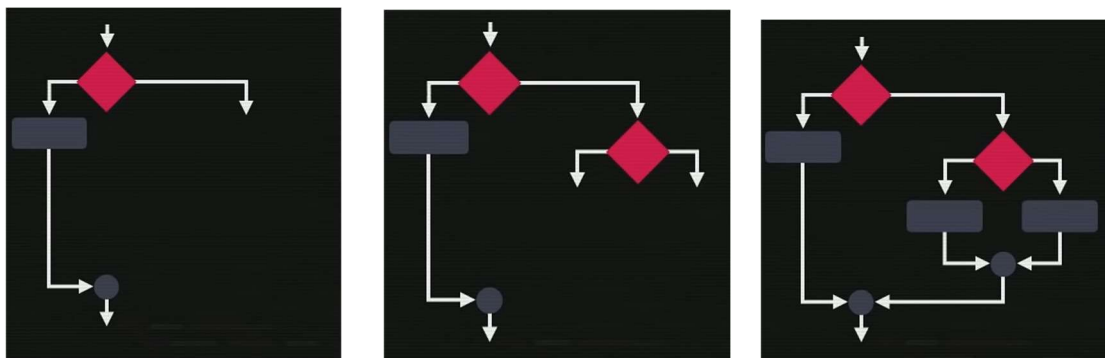
Agora sabemos que temos dois tipos de **condição**, a **condição simples** e a **condição composta**.

As condições simples são **ifs**, tendo uma condição, se ela for satisfeita vai executar um bloco, se eu precisar de blocos duplos, se uma determinada condição for verdadeira, faz alguma coisa, e se não faz outra coisa, tenho as condições compostas, os **elses**.



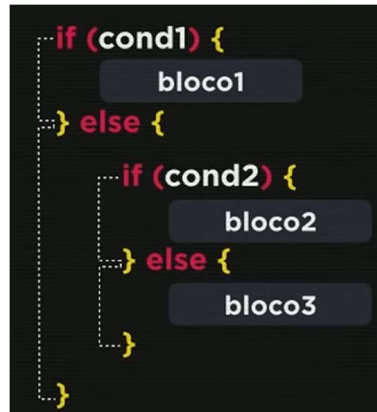
Existem situações em que existem várias possibilidades, ou faz isso, ou faz aquilo, ou faz aquilo outro, ou faz aquele, são várias formas de tratar um determinado dado.

E aí a condição simples e a condição composta, não servem de muita coisa. Mas existe uma possibilidade de criar, condições aninhadas, basta pegar uma condição composta e colocar outras condições dentro dela.



Por exemplo, na condição representada, se for verdade, faz o primeiro bloco, se não, não faço outro bloco, mas coloco outra condição e isso amplia muito as minhas possibilidades. Essa minha condição vai ter duas possibilidades, eu posso executar um bloco e fazer outra condição dentro e ir aninhando, essa palavra vem de ninho, um dentro do outro, se for verdade faço uma, se for falso eu faço outra, agora tenho três possibilidades, basta olhar a quantidade de blocos que temos agora, sendo assim quando tenho uma condição aninhada, tenho uma condição dentro da outra.

No JavaScript temos uma estrutura, um comando bem especial, que é o else-if.



Tenho aqui a minha condição principal, a condição 1, tenho um **if**, se a condição 1 for verdadeira, ele faz o bloco 1, se não, **else**, ele não vai fazer o bloco 2, seria uma condição composta, ele vai fazer outra condição, vai fazer o outro **if**, se a condição 2 for satisfeita, faço o bloco 2.

Importante que você perceba que, para executar o bloco 2, a condição 1 tem que ser **falsa**, porque se ela for **verdadeira**, ela já executa o bloco 1, se a condição 1 for **falsa** e a condição 2 for **verdadeira**, ele faz o bloco 2, se a condição 1 for **falsa** e a condição 2 for **falsa**, ele faz o bloco 3.

Nada impede que no lugar do bloco 3, colocar outra condição, assim estaria aumentando um nível no meu aninhamento.

Vamos abrir o nosso Visual Studio Code e vamos executar alguns códigos em JavaScript.

Quero dizer se uma pessoa vota ou não vota.

```
var idade = 16
if (idade < 16) {
    console.log('Não vota!')
} else {
    console.log('Vota')
}
```

Sabemos que no Brasil, uma pessoa com 16 anos ela vota, é voto opcional. Então vamos fazer o seguinte, como funciona o sistema de voto aqui no Brasil?

Quem tem abaixo de 16 realmente não vota, se não, eu tenho a opção do voto opcional e do voto obrigatório. Se a idade foi maior ou igual a 16 e a idade for menor do que 18, tenho um bloco aqui, que vai ser voto opcional.

```
var idade = 18
```

```

if (idade < 16) {
    console.log('Não vota!')
} else {
    if (idade < 18) {
        console.log('Voto opcional')
    }
}
}

```

Vamos testar, com 16 anos, vota opcional, 12 anos, não vota, 22 anos, o programa não mostrou nada, significa que o meu programa não deu erro sintático, ele deu erro de lógica, porque abaixo de 16, ele não vota, acima de 16 e menor do que 18, vota opcional, mas maior do que 18, não tem essa possibilidade no programa.

Dentro do JavaScript, como else e if é muito comum, posso pegar if e jogar dentro do else, economizo a escrita de um bloco, eu posso eliminar um bloco, se a idade for menor que 16 não vota, se não, se a idade for menor que 18, sabendo que menor que 18 maior ou igual a 16, o voto é opcional, se não, se a idade for maior ou igual a 18, o voto obrigatório.

```

var idade = 18
if (idade < 16) {
    console.log('Não vota!')
} else if (idade < 18) {
    console.log('Voto opcional')
} else {
    console.log('Voto obrigatório!')
}

```

Só que tem um outro pequeno problema no Brasil, quem tem acima de 65 anos, por exemplo 67 anos, o voto volta a ser opcional, e neste caso, ele diz que o voto é obrigatório.

Você não vai receber mensagens de erro, isso porque a lógica está certa, isso porque no Brasil, quem tem acima de 16 menor que 18 ou a idade acima de 65, o voto é opcional.

```

else if (idade < 18 || idade > 65)

```

Vou executar com 67 anos, o voto é opcional.

```

var idade = 18
if (idade < 16) {
    console.log('Não vota!')
} else if (idade < 18 || idade > 65) {

```

```
    console.log('Voto opcional')
} else {
    console.log('Voto obrigatório!')
}
```

Vou colocar um console.log, Você tem X anos.

```
var idade = 67
console.log(`Você tem ${idade} anos.`)
if (idade < 16) {
    console.log('Não vota!')
} else if (idade < 18 || idade > 65) {
    console.log('Voto opcional')
} else {
    console.log('Voto obrigatório!')
}
```

Vamos fazer outro exercício, quero dar bom dia, boa tarde e boa noite, e para isso, tenho que saber que horas são.

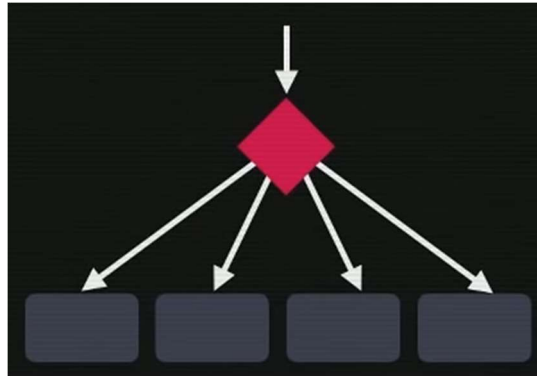
```
var hora = 8
console.log(`Agora são exatamente ${hora} horas.`)
if (hora < 12) {
    console.log('Bom Dia!')
} else if (hora <= 18) {
    console.log('Boa Tarde!')
} else {
    console.log('Boa Noite!')
}
```

Vamos pega a hora atual do sistema.

```
var agora = new Date()
var hora = agora.getHours()
console.log(`Agora são exatamente ${hora} horas.`)
if (hora < 12) {
    console.log('Bom Dia!')
} else if (hora <= 18) {
    console.log('Boa Tarde!')
} else {
    console.log('Boa Noite!')
}
```

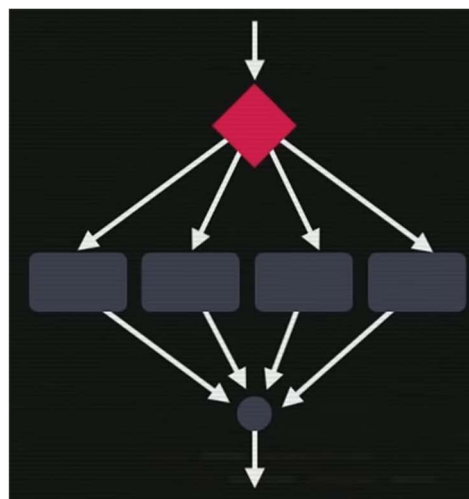
Com esse comando podemos pegar a hora atual do sistema que está rodando no script.

Além da **estrutura condicional alinhada**, existe uma outra muito importante, que é a **condição múltipla**, que serve para **valores fixos**. Ela é muito útil no mundo da programação, principalmente quando quero trabalhar valores fixos. Ela não serve muito para intervalos de valores, como a gente exemplificou ali o bom dia, boa tarde, boa noite.



Essa condição múltipla, tem a possibilidade não só do sim e não, ela tem a possibilidade de outros valores, de valores fixos.

Então, por exemplo, tenho aqui, se for um determinado valor, faz um bloco, se for outro valor, faz outro, se for outro valor, faz outro, e assim consigo representar esse tipo de estrutura. Tenho uma condição, uma expressão, e dela eu testo vários valores e depois eu volto para o fluxo normal do programa.



Esse tipo de **estrutura condicional** é a minha **condição múltipla**, ela não é aplicável para todo tipo de situação.

O if com else if, que a gente acabou de ver, é aplicável para todo tipo de situação, mas essa daqui é muito útil em situações pontuais específicas. E para representar essa estrutura, temos um comando dentro do JavaScript, que é o comando **switch**.

```
switch (expressão) {  
  
}
```

Switch, note que não é uma condição, é uma expressão, e tem um bloco relacionado, e vou colocar os valores dentro dela.

```
switch (expressão) {  
    case valor 1:  
  
    case valor 2:  
  
    case valor 3:  
  
    default:  
  
}
```

Tenho várias possibilidades de valor, para cada um vou colocar um **case**, se essa expressão for o primeiro valor, faz esse bloco, se for o segundo valor, faz o segundo, se for o terceiro, faz outro, e tem uma **cláusula** embaixo que é o **default**, que é como se fosse o **else** do switch, se nenhum dos valores de cima forem satisfeitos, ele vai fazer o de baixo.

```
switch (expressão) {  
    case valor 1:  
          
    case valor 2:  
          
    case valor 3:  
          
    default:  
          
}
```

E para cada um dos cases tenho um bloco. Basicamente a estrutura é dessa maneira, coloco um case para cada valor e por fim opcionalmente posso colocar um default, que é o padrão, que é se nenhum dos de cima forem satisfeitos.

Dentro da estrutura switch, Isso veio da linguagem C, existe um detalhe que precisa ser seguido, dentro de cada bloco, é preciso colocar um comando **break**.


```
switch (expressão) {  
  case valor 1:  
    [ ]  
    break  
  case valor 2:  
    [ ]  
    break  
  case valor 3:  
    [ ]  
    break  
  default:  
    [ ]  
    break  
}
```

Para cada um, tenho um **break**, no último é opcional, mas vamos colocar sempre, ele é obrigatório.

Ele funciona testando essa expressão.

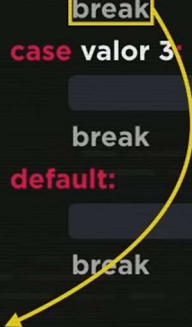
Vamos supor que essa expressão resulte no valor 2.

```
switch (expressão) {  
  case valor 1:  
    [ ]  
    break  
  case valor 2:  
    [ ]  
}
```



Ele vai desviar automaticamente, segue essa amarela, vai desviar para o comando 2, vai executar os comandos que estiverem nesse bloco, podem ser um ou vários comandos, e no final ele vai bater no break.

```
    break  
  case valor 3:  
    [ ]  
    break  
  default:  
    [ ]  
    break  
}
```



Quando ele bate no break, vai ser desviado lá para baixo. Se por acaso esse break não existir, vai dar problema, porque ele vai continuar executando todos os comandos até achar um break.

Vamos para o Visual Studio Code.

Vamos trabalhar com datas.

```
var agora = new Date()
var diaSem = agora.getDay()

/*
Domingo
Segunda
Terça
Quarta
Quinta
Sexta
Sábado
*/

console.log(diaSem) //comenta e coloca diaSem = 0 e testa

switch(diaSem) {
    case 0:
        console.log('Domingo')
        break
    case 1:
        console.log('Segunda')
        break
    case 2:
        console.log('Terça')
        break
    case 3:
        console.log('Quarta')
        break
```

```
case 4:
    console.log('Quinta')
    break
case 5:
    console.log('Sexta')
    break
case 6:
    console.log('Sábado')
    break
default:
    console.log('[ERRO] Dia Inválido!')
    break
}
```

E uma observação importante, não tem como dizer assim, se o dia da semana for entre 0 e 8, nesse caso, se você precisar testar intervalos, é muito mais valioso você utilizar o if.

O switch é uma estrutura muito importante para você testar dados pontuais, valores pontuais, 0, 1, 2, 3, não intervalos, ele só funciona com números inteiros e com caracteres, com strings.

O switch é uma estrutura um pouco mais limitada do que o if, mas ele é muito útil em situações pontuais, eles te ajudam bastante.