




Aula 1

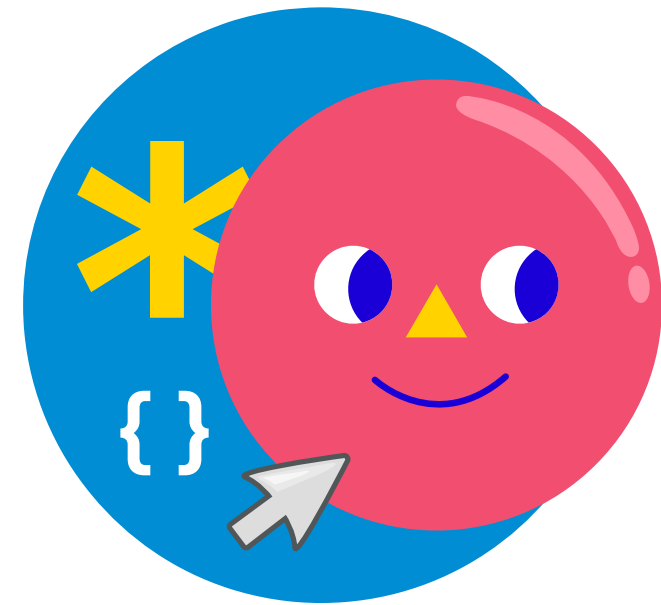
Iniciando nossa jornada

► **Unidade**

**Lógica de programação:
criando arte interativa com P5.js**

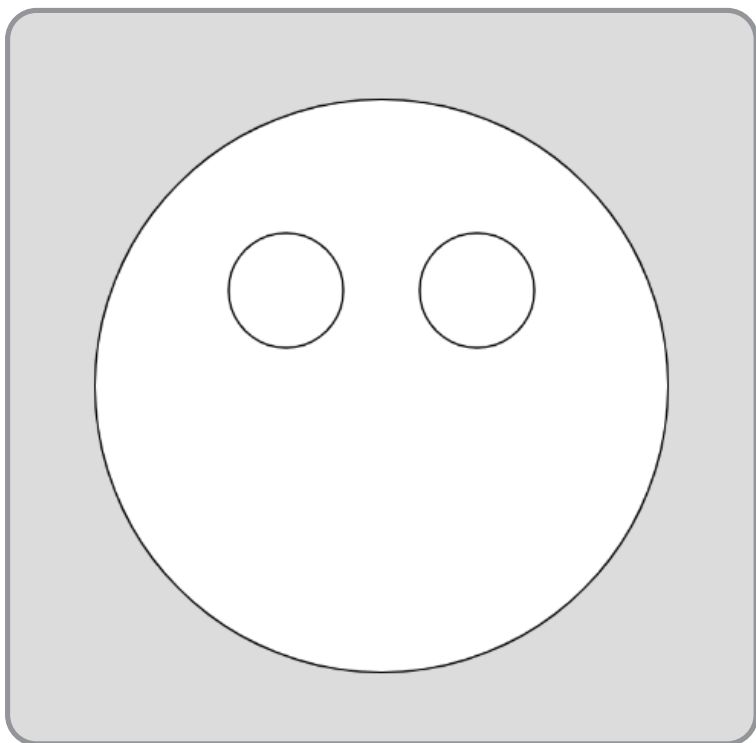
O que vamos aprender?

-  Compreender os conceitos básicos de funções em JavaScript.
-  Aplicar comandos básicos de desenho no P5.js, como **createCanvas()** e **circle()**.
-  Modificar parâmetros de comandos em P5.js para ajustar a posição e o tamanho de formas geométricas.



ACOMPANHE O VÍDEO DA AULA

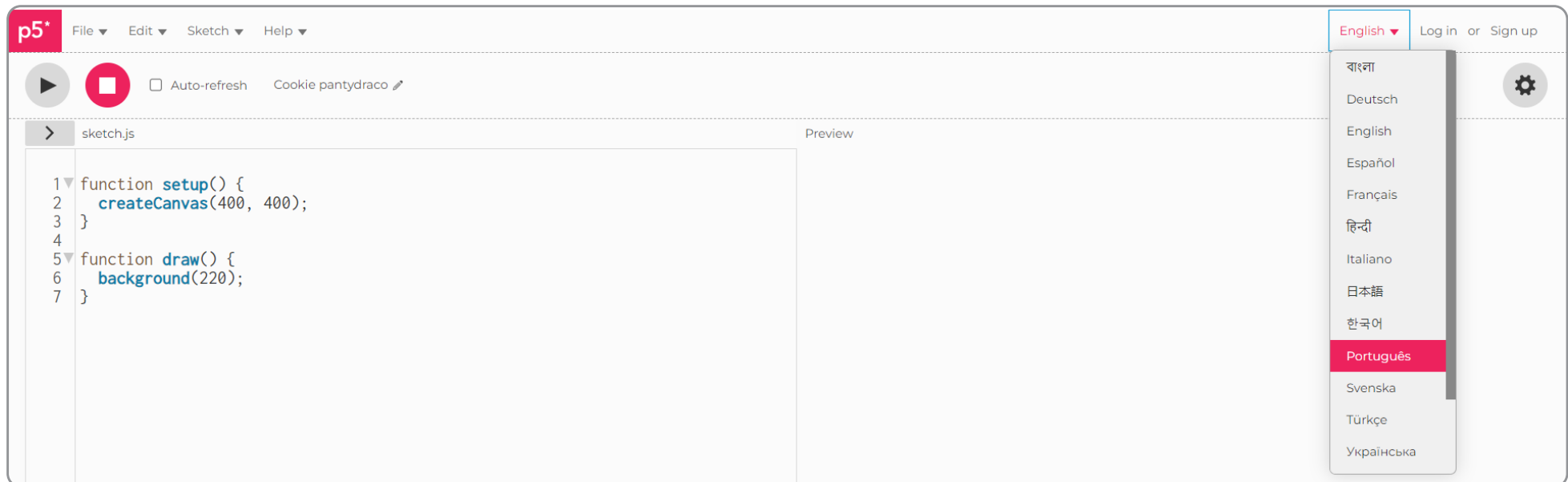
Primeiras linhas de código



Nesta unidade, construiremos dois projetos: o Monalisa e o Quente e frio. O projeto Monalisa trabalhará com a ideia de uma forma que segue o ponteiro do mouse, enquanto o projeto Quente e frio abordará números aleatórios. Para isso, utilizaremos a ferramenta P5.js e a linguagem JavaScript. Nesta primeira aula, trabalharemos com o desenho e o posicionamento de formas geométricas, iniciando o projeto Monalisa.

Começaremos o projeto acessando o site do P5.js através do link <https://editor.p5js.org/>. Observe que, antes de desenvolver o projeto, é necessário criar uma conta para que você possa salvar os códigos criados e retornar a eles sempre que necessário.

Para isso, acesse o site indicado e, para facilitar o processo, altere o idioma para português no menu de idiomas, localizado no canto superior direito da tela:



Ao fazer isso, você verá, ao lado do menu de idiomas, as opções *Entrar* ou *Registrar-se*. **Clique em *Registrar-se*.**

Português ▼ Entrar ou Registrar-se

Em seguida, uma nova tela se abrirá solicitando informações para a criação de uma conta. Forneça essas informações e clique em *Entrar*, ou entre com sua conta Google ou do GitHub, se preferir.

Criar Conta

Nome de Usuário


Email


Senha

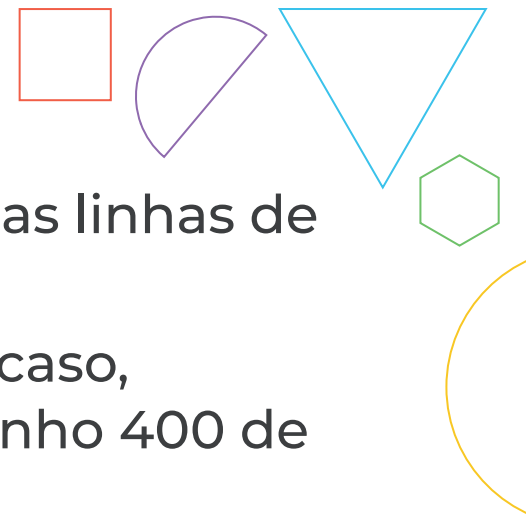
Confirmar Senha

Entrar

Ou

 Entrar com GitHub

 Entrar com Google

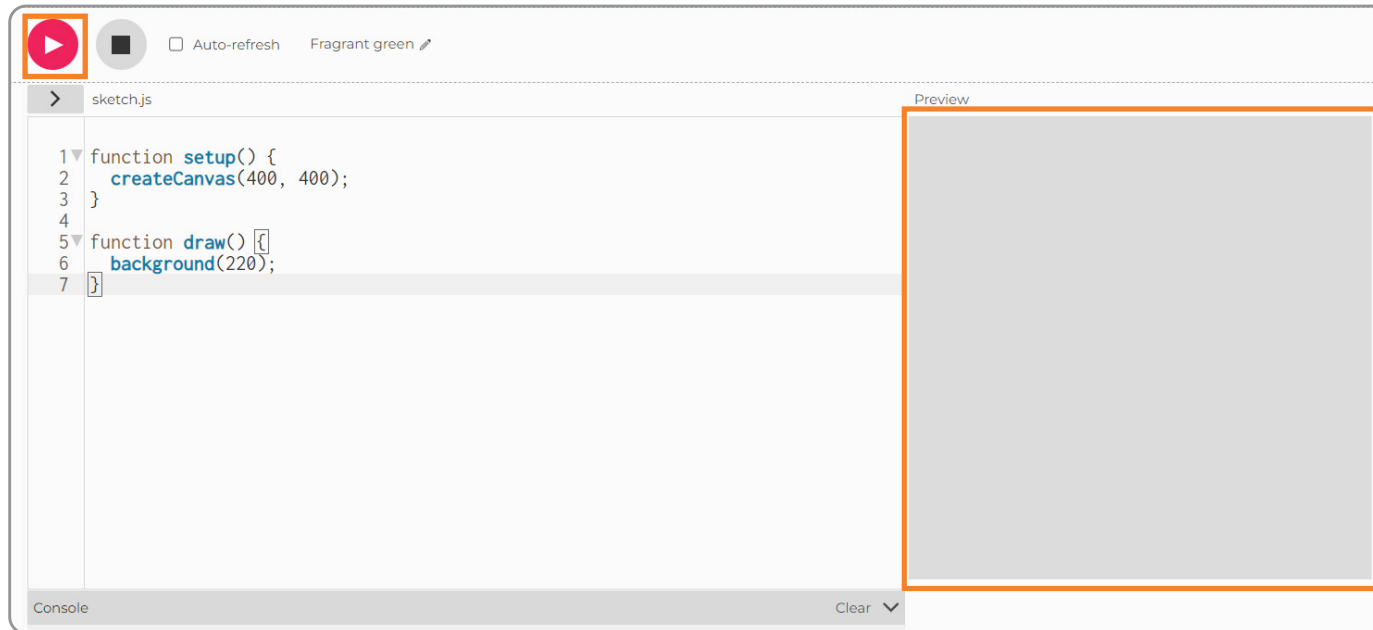


Logo de início, ao acessar a ferramenta, ela já mostra algumas linhas de código. Vamos compreendê-las!

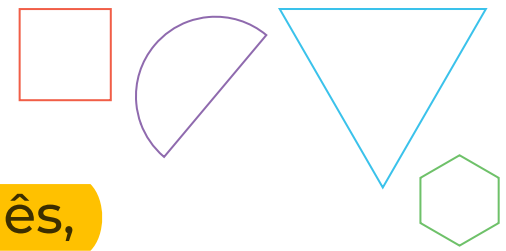
A função **setup** serve para inicializar alguma coisa e, nesse caso, estamos inicializando um canvas, ou seja, uma tela de tamanho 400 de largura por 400 de altura.

```
function setup() {  
  createCanvas(400, 400);  
}  
  
function draw() {  
  background(220);  
}
```

Podemos clicar no botão de executar, no canto superior esquerdo da tela, para ver como esse canvas aparece. Confira na imagem:

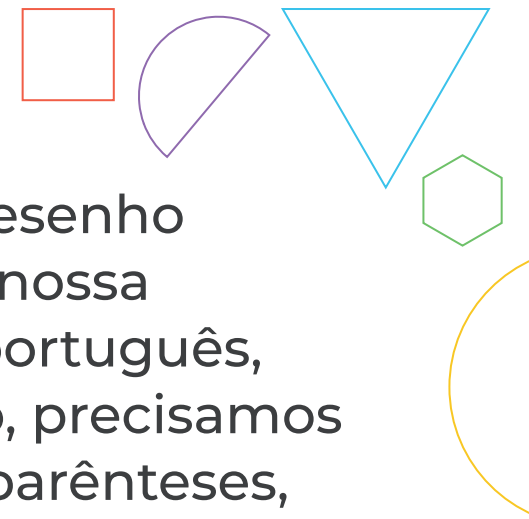


O quadrado cinza na área de Preview é o nosso canvas. Podemos alterar os valores dentro da função **createCanvas** para torná-lo maior ou menor.



Abaixo da função **setup**, temos a função **draw** (em português, **desenhar**). Todos os desenhos que criaremos precisam ficar dentro dessa função. Observe que já está sendo passada a função **background**, que é responsável por alterar a cor de fundo do canvas. Nesse caso, o valor **220** representa a cor cinza e, assim como na função anterior, podemos alterar esse valor e ver as cores que aparecem.

```
function setup() {  
  createCanvas(400, 400);  
}  
  
function draw() {  
  background(220);  
}
```

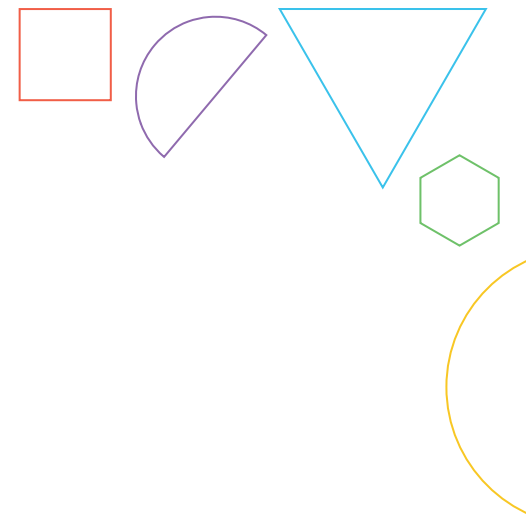



Tendo compreendido esses comandos iniciais, o primeiro desenho que criaremos será um círculo, que representará o rosto da nossa personagem. Para isso, utilizaremos a função **circle** (em português, círculo). Como você deve ter percebido, ao criar uma função, precisamos adicionar parênteses logo após seu nome e, dentro desses parênteses, vamos inserir alguns valores.

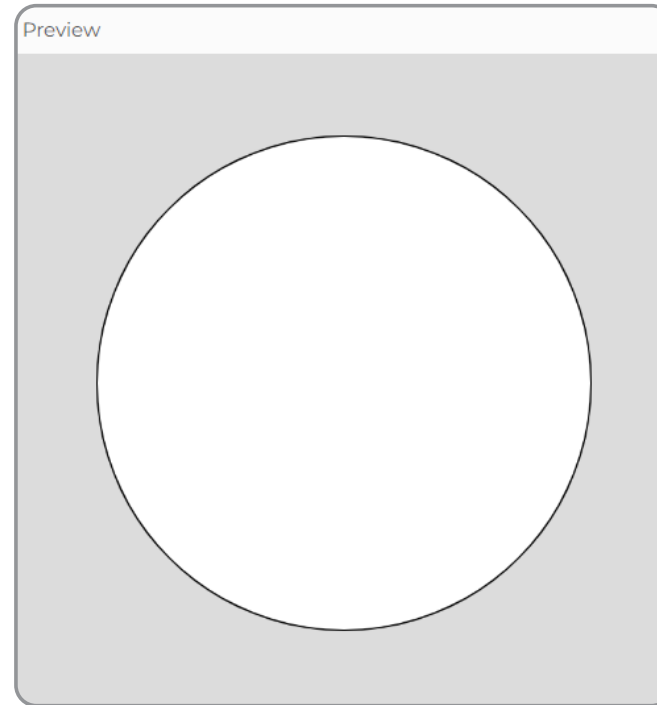
Para que a função **circle** funcione, precisamos de três valores: sua posição no eixo X, sua posição no eixo Y e o diâmetro do círculo. Confira o código:

```
function setup() {  
  createCanvas(400, 400);  
}  
  
function draw() {  
  background(220);  
  circle(200, 200, 300);  
}
```

Os valores **200** e **200**, para os eixos X e Y, respectivamente, representam o centro do nosso canvas. O valor **300** representa o diâmetro do círculo.

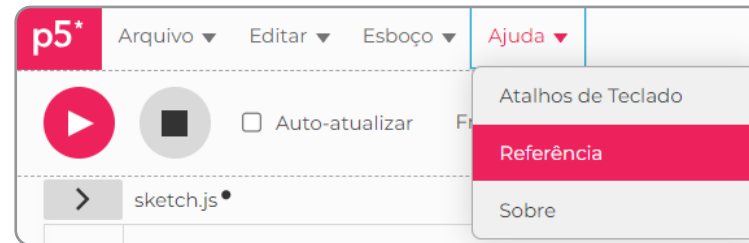


Ao executar o código, o resultado será o seguinte:

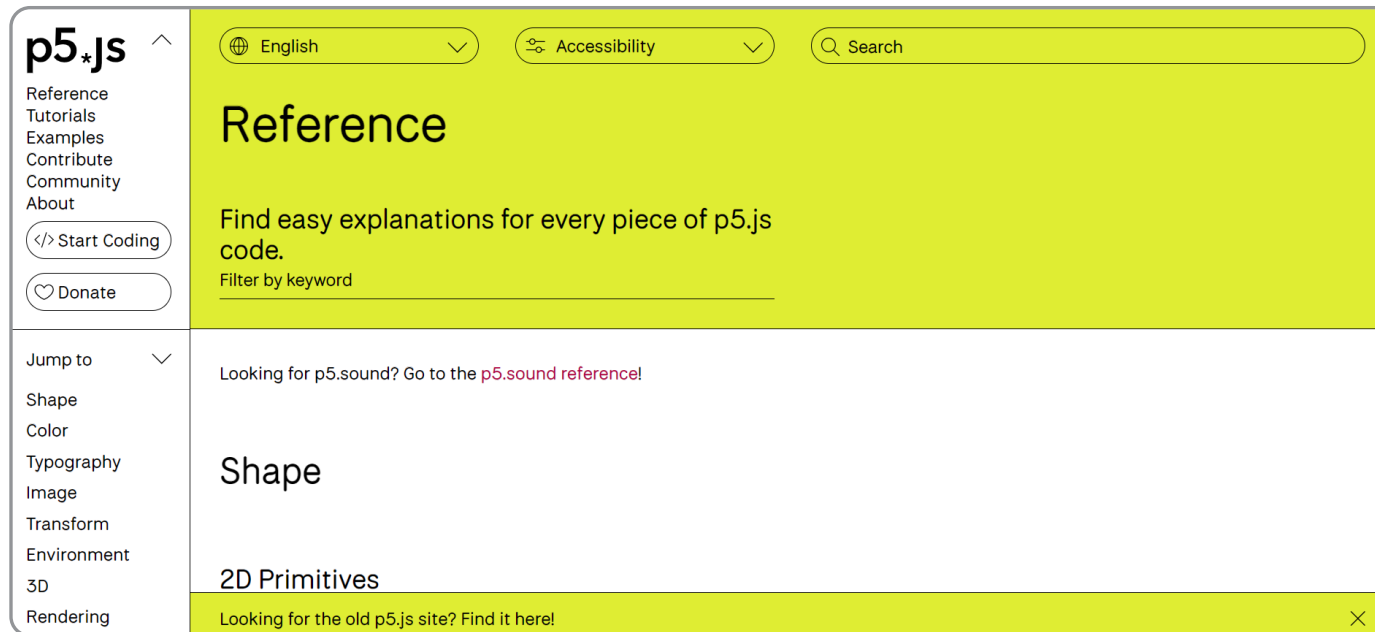


O círculo aparece corretamente. Mas, e se quisermos adicionar outras formas? Como sabemos quais funções precisamos usar? Vamos descobrir!

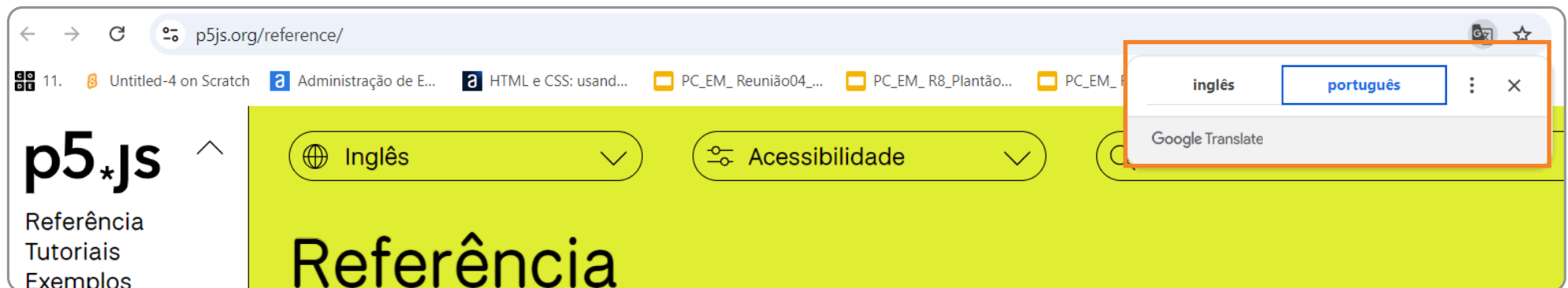
Observe que, no canto superior direito da página do P5.js, há um menu chamado *Ajuda*. Nele, acessaremos a opção *Referência*, conforme imagem abaixo:

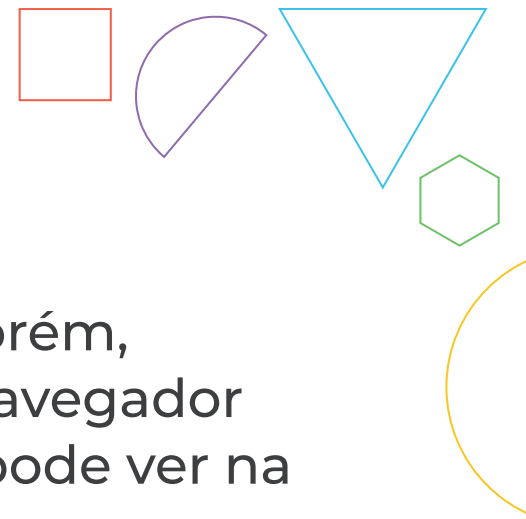


Ao clicar nessa opção, seremos redirecionados para a seguinte página:



Veja que a página está toda em inglês e, nas opções de alterar idioma, o português não está disponível. Nesse caso, podemos usar o recurso de tradução do próprio navegador. Caso você esteja usando o Google Chrome, esse recurso estará disponível no final da barra de endereço, no lado direito da tela:





Ao selecionar essa opção, a página inteira será traduzida. Porém, precisamos tomar cuidado, pois o recurso de tradução do navegador traduz até mesmo o nome das nossas funções, como você pode ver na imagem abaixo:

Primitivos 2D

`arco()`

Desenha um arco.

`círculo()`

Desenha um círculo.

`elipse()`

Desenha uma elipse (oval).

`linha()`

Desenha uma linha reta entre dois pontos.

`apontar()`

Desenha um único ponto no espaço.

`quadrângulo()`

Desenha um quadrilátero (forma de quatro lados).

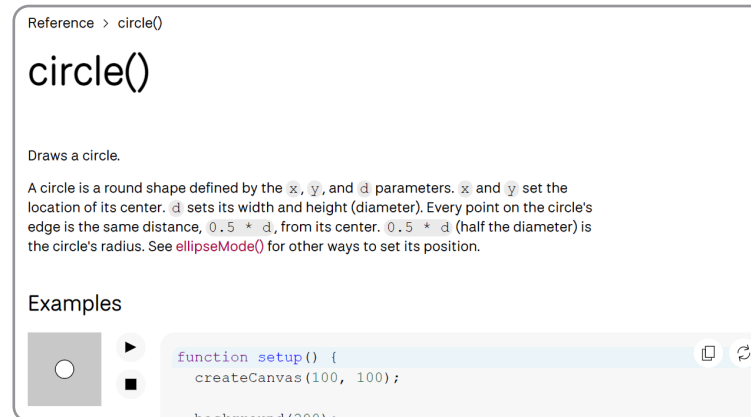
`reto()`

Desenha um retângulo.

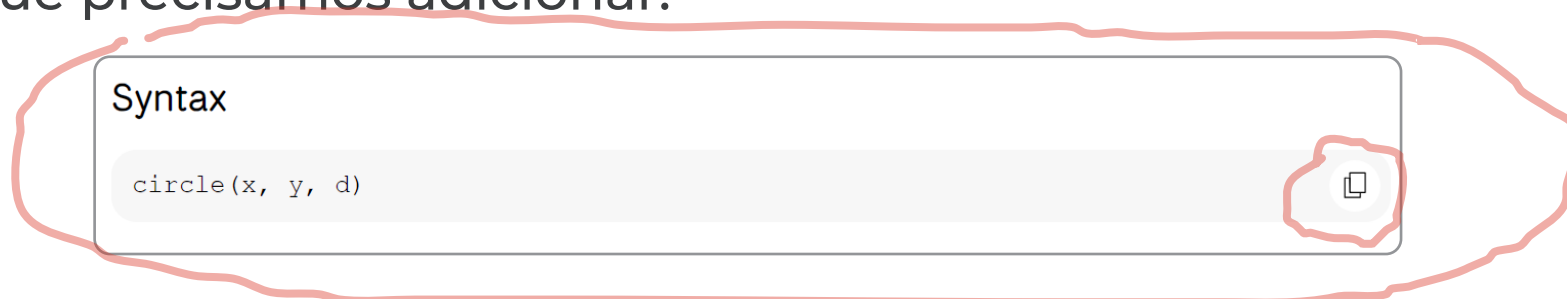
`quadrado()`

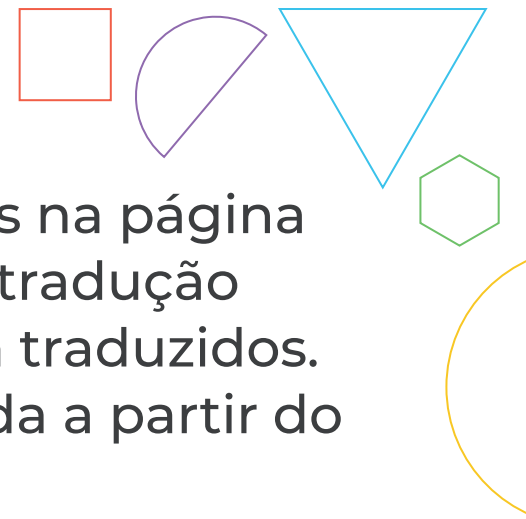
Desenha um quadrado.

A função **círculo()** não existe no P5.js, pois os nomes de função são todos em inglês. Desse modo, precisamos tomar cuidado para que a tradução não gere erros ao programar. Para verificar o nome correto de cada função, basta acessá-la clicando sobre o seu nome. No caso da função **circle**, a seguinte página se abrirá ao clicarmos sobre ela:



Rolando a página para baixo, teremos acesso à sintaxe da função, que nada mais é do que a maneira correta de escrevê-la no P5.js, com os valores que precisamos adicionar:





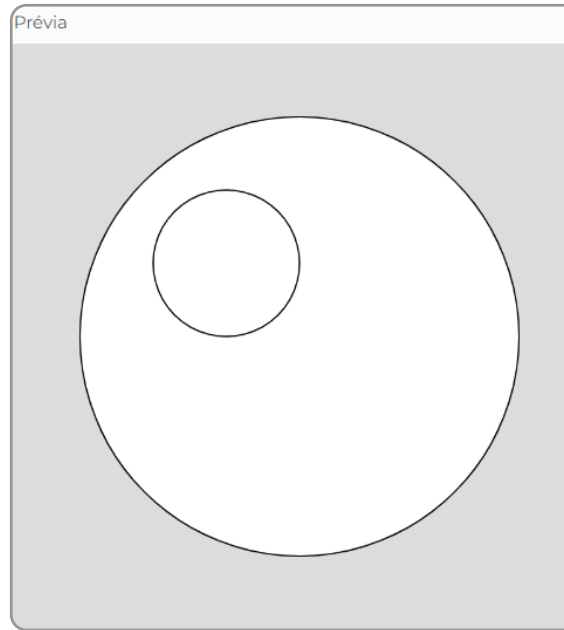
É importante que você explore as outras funções disponíveis na página de referência. Porém, tome cuidado ao utilizar o recurso de tradução do navegador para que os códigos em JavaScript não sejam traduzidos. Lembre-se de que essa linguagem de programação foi criada a partir do inglês e, portanto, não pode ser traduzida.

Agora, continuaremos a criar nossa obra de arte adicionando um olho! Para isso, utilizaremos a função **circle** novamente, porém, com valores diferentes. Confira o código:

```
function draw() {  
  background(220);  
  circle (200,200,300);  
  circle (150,150,100);  
}
```

Repare que modificamos os valores de x e y para **150**, de modo que esse segundo círculo apareça em uma posição diferente do centro.

Ao executar o código, temos o seguinte resultado:



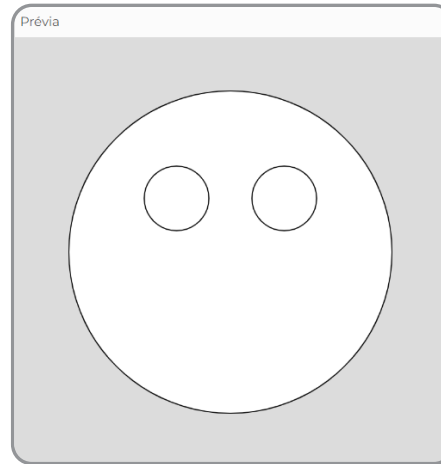
Observe que o olho está muito grande! Precisamos mudar o valor do diâmetro do círculo para corrigir esse problema. Para isso, podemos utilizar o valor 60 ou outro que você desejar. Além disso, vamos adicionar novamente a função **circle** para criar o outro olho, do lado direito. Nesse caso, precisamos também mudar o valor no eixo X.

O código para adicionar dois olhos ficará assim:

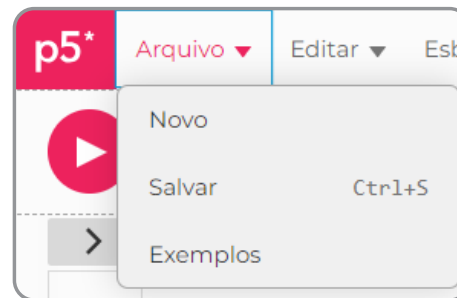
```
function draw() {  
  background(220);  
  circle (200,200,300);  
  circle (150,150,60);  
  circle (250,150,60);  
}
```

Perceba que o valor do eixo Y (**150**) não foi alterado, pois queremos que os dois olhos tenham a mesma altura. Entretanto, fique à vontade para testar valores diferentes e construir sua obra de arte como desejar.

O resultado será o seguinte:



Antes de finalizar, lembre-se de salvar o projeto acessando o botão *Salvar*, no menu *Arquivo*, localizado no canto superior esquerdo da tela, ou utilizando o atalho *Ctrl+S*.



Nas próximas aulas, daremos seguimento ao projeto adicionando novas formas e explorando novas funções!

► Desafio

Nesta aula, conhecemos a ferramenta P5.js e iniciamos um projeto artístico com figuras geométricas adicionadas através de funções. Seu desafio será explorar a página de referências e escolher uma nova função, que adiciona uma figura geométrica diferente. Tente implementar essa função no seu projeto e esteja atento a possíveis erros de tradução caso utilize a ferramenta de tradução automática do navegador. Lembre-se: em JavaScript, todas as funções devem ser construídas em inglês.

