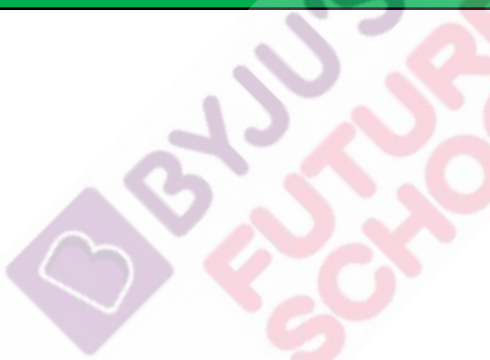
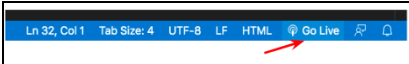


Tópico	JOGO COM IA	
Descrição da Aula	A criança revisa e reaplica todos os conceitos de <i>machine learning</i> (aprendizado de máquina) para aprimorar um jogo popular, fazendo com que funcione com redes neurais. Assim, aprenderá a aplicar IA no mundo real.	
Aula	ADV-C139	
Duração da Aula	55 min	
Objetivo	<ul style="list-style-type: none"> • Completar a GUI do jogo com a adição da class Bootstrap modal 	
Recursos Necessários	<ul style="list-style-type: none"> • Recursos da Professora <ul style="list-style-type: none"> ○ Suas credenciais de login do Gmail; ○ Fone de ouvido com microfone; ○ Bloco de notas e caneta. • Recursos do Aluno <ul style="list-style-type: none"> ○ Suas credenciais de login do Gmail; ○ Fone de ouvido com microfone (opcional); ○ Bloco de notas e caneta. 	
Estrutura da Aula	Introdução Atividade Dirigida pela Professora Atividade Dirigida pelo Aluno Encerramento	5 min 15 min 35 min 5 min
Passos da Aula	Dizer	Fazer
Passo 1: Introdução (5 min)	<p>Na última aula, iniciamos o desenvolvimento do Jogo Mario com IA, e adicionamos alguns códigos HTML e CSS.</p> <p>Como combinamos na última aula, você não deveria modificar os códigos da pasta Mario-Game.</p>	<p>Peça para o aluno ativar o modo Tela cheia.</p>

	<p>Na aula de hoje, completaremos os código HTML e CSS e a GUI do jogo com a adição do Bootstrap modal.</p> <p>P: Você se lembra do Bootstrap modal? Em qual atividade utilizamos ele?</p> <p>R: O Bootstrap modal é um pop-up. Utilizamos ele quando desenvolvemos nosso blog pessoal pela aula 68.</p>	
A professora inicia o compartilhamento de tela		
<p>Passo 2: Atividade Dirigida pela Professora (15 min)</p>		<p>Se você fizer o download do código fonte completo da Atividade da Professora 3 e quiser executá-lo em seu sistema, CERTIFIQUE-SE DE FAZER ISSO CLICANDO NO BOTÃO GO LIVE DO VISUAL STUDIO. COM ISSO, OS ARQUIVOS SERÃO EXECUTADOS NO LIVE SERVER DO VISUAL STUDIO</p>  <p>Utilizaremos arquivos de áudio e imagem, porém, a biblioteca p5.js não nos permite executar esses tipos de arquivos do sistema local, portanto, precisaremos executá-los em um SERVER.</p> <p>Por favor, siga o fluxo para esta aula:</p> <p>Explique os códigos HTML e CSS.</p>

		<p>Deixe o aluno escrever os códigos HTML e CSS.</p> <p>Continue com as Atividades Adicionais se houver tempo.</p>
--	--	--

Na última aula, concluímos:

OBSERVAÇÃO: A imagem abaixo está presente na **Atividade da Professora 2** para representação visual. Por favor, abra-a enquanto explica:

```
<body background="background.jpg">
  <center>
    <div class="btn btn-primary heading">
      <h3>IA MARIO</h3>
      <button class="btn btn-primary" data-toggle="modal" data-target="#myModal">Instruções</button>
    </div>
    
    <br><br>
    <button class="btn btn-success" onclick="startGame()" id="start">Jogar</button>
    <br><br>
    <h3 id="status" class="btn btn-warning"></h3>
    <br><br>
    <div id="canvas"></div>
    <div id="gameConsole"></div>
  </center>
</body>
```

Adicionamos:

1. Uma imagem de fundo para a página web;
2. Um título e um botão de instruções;
3. Uma imagem grande do Mario;
4. Um botão “Jogar” e um elemento HTML vazio para conter o status do jogo;
5. Duas divs vazias para conter o canvas e a visualização da webcam.

Continuaremos o código HTML no arquivo [index.html](#) e o CSS em [style.css](#)

1. Primeiro, adicionaremos uma tag h4 para conter a seção de créditos. Nela, adicionaremos o link github de Linuk, que criou o código do jogo, e a menção ao jogo Mario.

OBSERVAÇÃO: Você pode copiar e colar esse link no chat para o aluno.

```
<body background="background.jpg">
  <center>

    <div class="btn btn-primary heading">
      <h3>IA MARIO</h3>
      <button class="btn btn-primary" data-toggle="modal" data-target="#myModal">Instruções</button>
    </div>

    
    <br><br>
    <button class="btn btn-success" onclick="startGame()" id="start">Jogar</button>
    <br><br>
    <h3 id="status" class="btn btn-warning"></h3>
    <br><br>
    <div id="canvas"></div>
    <div id="gameConsole"></div>
    <h4>Código Fonte: <a href="https://github.com/linuk">Linuk</a> || Créditos: Mario</h4>
  </center>
```

- Na primeira parte, escreveremos “Código fonte:”
- Adicionaremos uma tag anchor e, dentro dela, mencionaremos uma href com o link GitHub de Linuk. Ao lado escreveremos “Linuk”

```
<h4>Código Fonte: <a href="https://github.com/linuk">Linuk</a>
```

- Agora, adicionamos o texto de crédito ao jogo Mario.

```
<h4>Código Fonte: <a href="https://github.com/linuk">Linuk</a> || Créditos: Mario</h4>
```

Estilo em **style.css**

```
h4
{
  background-color: #f5f5f5;
  line-height: 2;
  margin: 0px;
}
```

- Definimos a cor de fundo para #f5f5f5, que é uma tonalidade de braco. Você pode utilizar

a cor que quiser;

- Definimos **line-height :2** para aumentar a altura da tag h4;
- Definimos margin para 0px, a fim de gerar um espaço entre esse elemento h4 e o fim da tela.

Saída:

```
<body background="background.jpg">
  <center>

    <div class="btn btn-primary heading">
      <h3>IA MARIO</h3>
      <button class="btn btn-primary" data-toggle="modal" data-target="#myModal">Instruções</button>
    </div>

    
    <br><br>
    <button class="btn btn-success" onclick="startGame()" id="start">Jogar</button>
    <br><br>
    <h3 id="status" class="btn btn-warning"></h3>
    <br><br>
    <div id="canvas"></div>
    <div id="gameConsole"></div>
    <h4>Código Fonte: <a href="https://github.com/linuk">Linuk</a> || Créditos: Mario</h4>
  </center>
```



2. Em seguida, adicionaremos uma div. Essa será a div principal, para conter o Bootstrap modal (pop-up). Para isso, utilizaremos muitas classes Bootstrap. Utilizar classes Bootstrap é muito vantajoso, pois precisamos apenas escrever o nome da class e seu estilo é automaticamente aplicado.

```
<body background="background.jpg">
<center>

  <div class="btn btn-primary heading">
    <h3>IA MARIO</h3>
    <button class="btn btn-primary" data-toggle="modal" data-target="#myModal">Instruções</button>
  </div>

  
  <br><br>
  <button class="btn btn-success" onclick="startGame()" id="start">Jogar</button>
  <br><br>
  <h3 id="status" class="btn btn-warning"></h3>
  <br><br>
  <div id="canvas"></div>
  <div id="gameConsole"></div>
  <h4>Código Fonte: <a href="https://github.com/linuk">Linuk</a> || Créditos: Mario</h4>

  <div id="myModal" class="modal fade ">
  </div>

</center>
```

- Defina a div e a ID para “myModal”. Lembre-se, é obrigatório que esse nome de id seja o mesmo passado dentro do atributo **data-target**, que mencionamos no botão “Instruções” da última aula.
- Em seguida, escreva a class Bootstrap **modal**: Ela nos ajuda a reduzir a visibilidade do fundo quando o pop-up aparece.
- Além disso, escreva a class Bootstrap “**fade**”: fade é a class que adiciona o efeito de animação a esse pop-up ao ser fechado.

3. Adicione mais uma div, dentro dessa div principal de modal, e escreva a class Bootstrap **modal-dialog**. Essa class adiciona preenchimento ao pop-up, para que o modal não toque as extremidades da tela.

```
<div id="myModal" class="modal fade ">
  <div class="modal-dialog" >
  </div>
</div>
```

4. Adicionaremos estilo a essa **modal-dialog** no arquivo **style.css**.

```
.modal-dialog
{
  width: 90%;
}
```

- Aumentamos a largura do modal definindo a *width* para 90%. Ou seja, ele cobrirá 90% da tela.
5. Em seguida, adicione mais uma div e escreva a class Bootstrap **modal-content**. Essa class adiciona a cor branca ao fundo, cor preta à fonte, *box-shadow* (sombra de caixa) e *border-radius* (borda arredondada) ao modal(pop-up), para que ele tenha uma aparência melhor. Essa div conterá todo o conteúdo do pop-up.

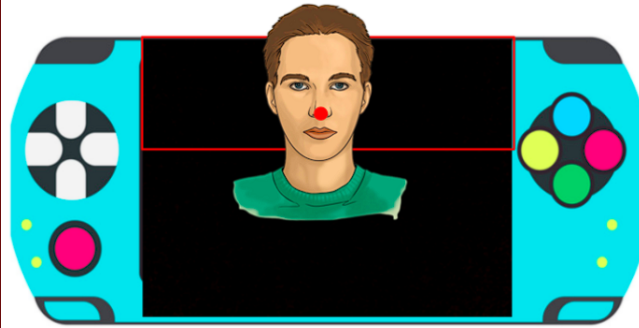
```
<div id="myModal" class="modal fade ">
  <div class="modal-dialog" >
    <div class="modal-content">
    </div>
  </div>
</div>
```

- Se não adicionarmos a class **modal-content**, então, teremos uma saída com aparência simples:

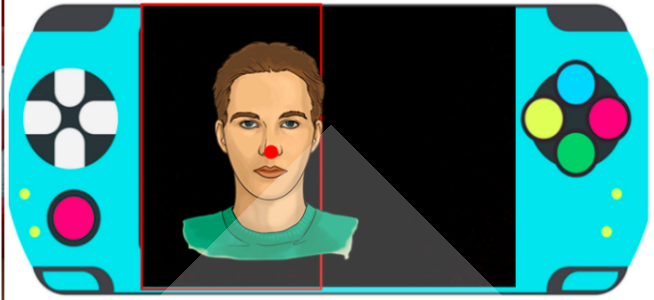
OBSERVAÇÃO: A imagem abaixo está presente na **Atividade da Professora** para representação visual. Por favor, abra a **Atividade da Professora 2** enquanto explica.

IA MARIO

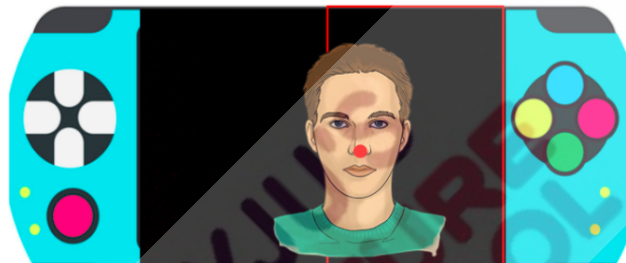
Coloque seu nariz na parte superior do vídeo para pular



Coloque seu nariz no lado esquerdo do vídeo para mover para a direita, pois a visualização da webcam não é uma imagem espelhada



Coloque seu nariz no lado direito do vídeo para mover para a esquerda, pois a visualização da webcam não é uma imagem espelhada



Instruções

- Em seguida, adicione mais uma div para conter a parte de **head** do pop-up, e forneça a class Bootstrap **modal-header**. Essa class adiciona *padding* (preenchimento) a head do modal, para que tenha uma aparência melhor. Essa conterá o título “Instruções” para o pop-up e um botão **X**, que será utilizado para fechá-lo.



```
<div id="myModal" class="modal fade ">
  <div class="modal-dialog" >
    <div class="modal-content">
      <div class="modal-header">
      </div>
    </div>
  </div>
</div>
```



- Dentro dessa div, adicionaremos primeiro um botão () para fechar o pop-up.


```
<div id="myModal" class="modal fade ">
  <div class="modal-dialog" >
    <div class="modal-content">
      <div class="modal-header">
        <button class="close" data-dismiss="modal">&times;</button>
      </div>
    </div>
  </div>
</div>
```

- Defina a tag *button* e forneça a class **close**. Com isso, modificamos o estilo padrão do

botão de  para .

- Em seguida, escreva **data-dismiss="modal"** para adicionar a funcionalidade de fechar a janela quando o botão for pressionado.

- Para obter o ícone X de Bootstrap , precisamos escrever **×** no texto do botão.

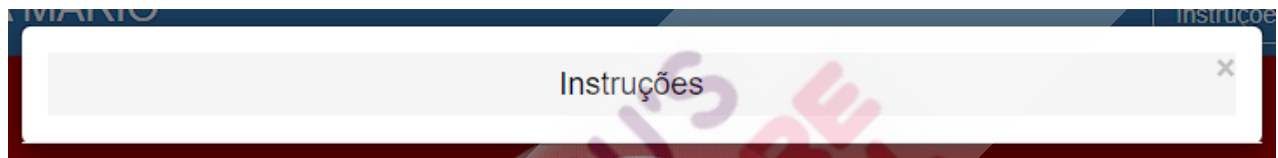
Saída quando clicamos no botão “Instruções”:



- Adicionaremos uma tag **h4** para conter o texto “Instruções”.

```
<div id="myModal" class="modal fade ">
  <div class="modal-dialog" >
    <div class="modal-content">
      <div class="modal-header">
        <button class="close" data-dismiss="modal">&times;</button>
        <h4>Instruções</h4>
      </div>
    </div>
  </div>
</div>
```

Saída quando clicamos no botão “Instruções”:



- Adicione mais uma div para conter a parte do **corpo** do pop-up, e forneça a class Bootstrap **modal-body**, a fim de adicionar *padding* e melhorar sua aparência. Essa parte conterá três imagens de instruções sobre como jogar.

OBSERVAÇÃO: A imagem abaixo está presente na [Atividade da Professora 2](#) para representação visual. Por favor, abra a [Atividade da Professora 2](#) enquanto explica.

```
<div id="myModal" class="modal fade ">
  <div class="modal-dialog" >
    <div class="modal-content">
      <div class="modal-header">
        <button class="close" data-dismiss="modal">&times;</button>
        <h4>Instruções</h4>
      </div>
      <div class="modal-body">
      </div>
    </div>
  </div>
</div>
```

8. Adicione a primeira imagem de instrução sobre como pular.

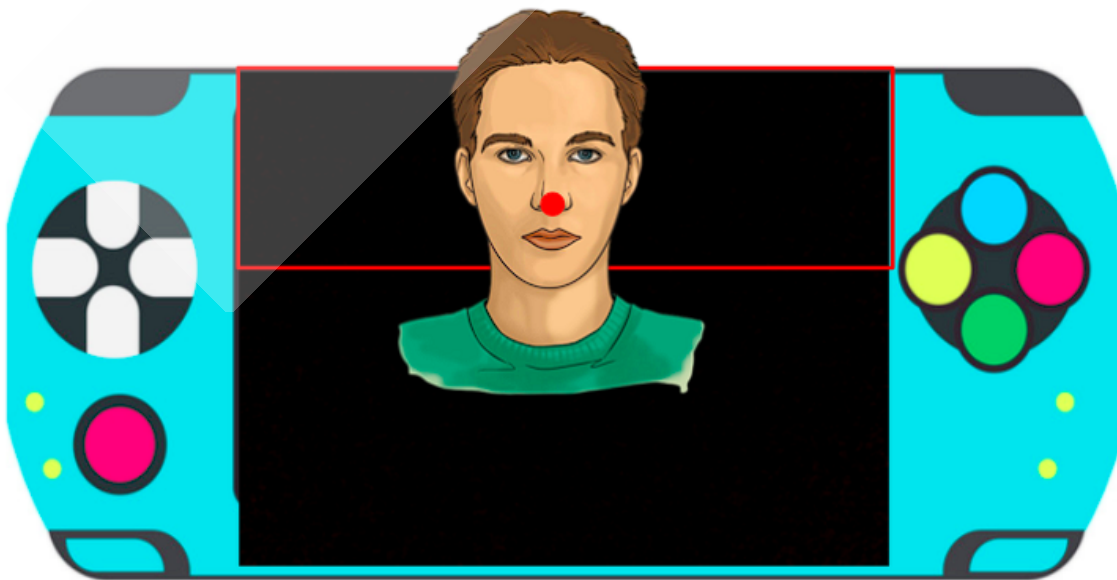
```
<div id="myModal" class="modal fade ">
  <div class="modal-dialog" >
    <div class="modal-content">
      <div class="modal-header">
        <button class="close" data-dismiss="modal">&times;</button>
        <h4>Instruções</h4>
      </div>

      <div class="modal-body">
        
      </div>
    </div>
  </div>
</div>
```

- Em src, mencione o nome da imagem **jump.png**. Coloque o mesmo nome da imagem presente na pasta **Mario-Game**;
- Adicione a class **instructionImage**: adicionaremos estilos a essa class em **style.css**. Mas, antes, adicionaremos mais duas imagens de instrução;
- Além disso, forneceremos a mesma class "**instructionImage**" a essas outras duas imagens, para que possamos escrever as propriedades de estilo apenas uma vez.

Saída:

Coloque seu nariz na parte superior do vídeo para pular



9. Adicione a segunda imagem de instruções sobre como mover para a direita, e forneça a class “**instructionImage**”.

```
</div>  
  
<div class="modal-body">  
    
  
```

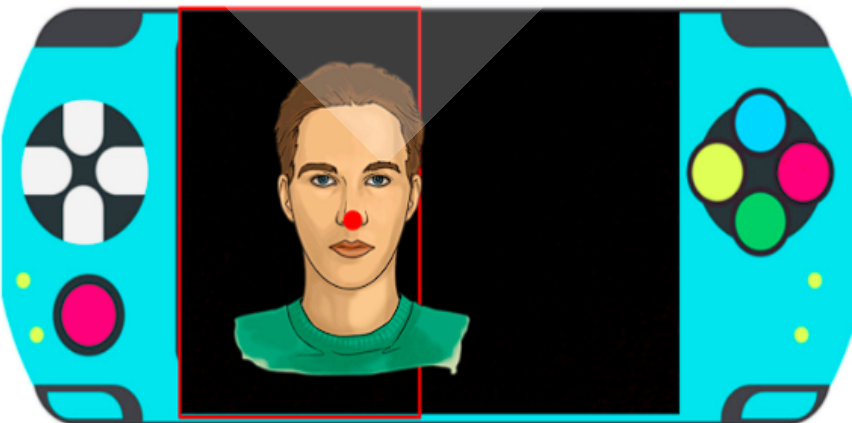
Saída:

Instruções

Coloque seu nariz na parte superior do vídeo para pular



Coloque seu nariz no lado esquerdo do vídeo para mover para a direita, pois a visualização da webcam não é uma imagem espelhada



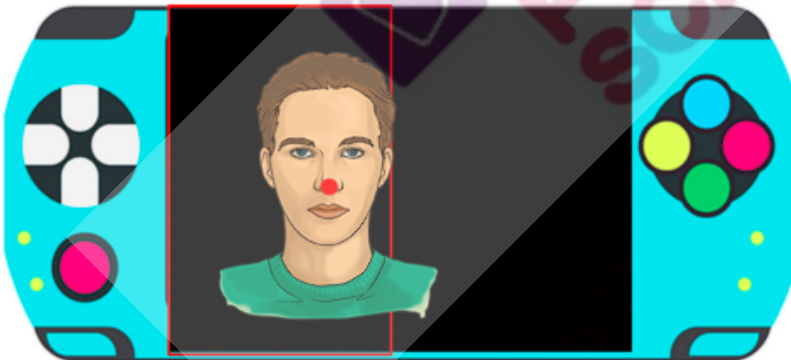
10. Adicione a terceira imagem sobre como se mover para a esquerda, e forneça a class "instructionImage".

```
<div class="modal-body">  
    
    
    
</div>
```

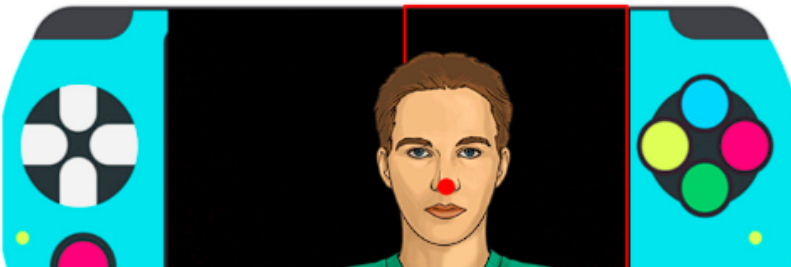
Coloque seu nariz na parte superior do vídeo para pular



Coloque seu nariz no lado esquerdo do vídeo para mover para a direita, pois a visualização da webcam não é uma imagem espelhada



Coloque seu nariz no lado direito do vídeo para mover para a esquerda, pois a visualização da webcam não é uma imagem espelhada



Saída:

11. Você pode ver na saída que todas as imagens estão em uma linha horizontal, o que não mostra um bom design. Portanto, para melhorar, forneceremos estilos a essa class “**instructionImage**”.

```
.instructionImage
{
  display: inline-block;
  width: 45%;
}
```

- Primeiro, precisamos escrever **display:inline-block**: esse código colocará todas as imagens em uma linha;
- Em seguida, precisamos definir a *width* (largura) para 45%, ou seja, a imagem será sempre 45% da tela.
- Duas imagens com *width* 45% significa que ocuparão 90% da tela, logo, não haverá espaço para a terceira e esta será empurrada para a segunda linha.

Saída:

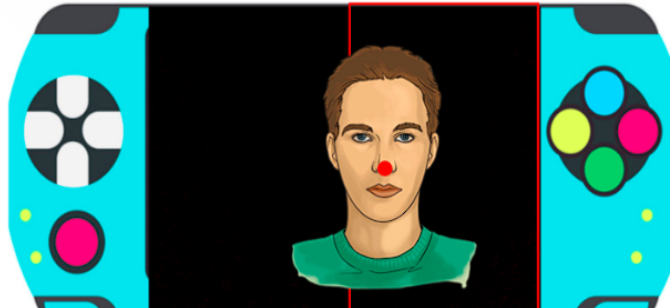
Instruções

Coloque seu nariz na parte superior do vídeo para pular

Coloque seu nariz no lado esquerdo do vídeo para mover para a direita, pois a visualização da webcam não é uma imagem espelhada



Coloque seu nariz no lado direito do vídeo para mover para a esquerda, pois a visualização da webcam não é uma imagem espelhada



Finalizamos o código HTML e um pouco do código CSS ainda está pendente, mas o adicionaremos na próxima aula.

Hoje, iniciaremos um pouco do código JS. Portanto, quando você for programar, precisará abrir o arquivo [main.js](#) e seguir minhas instruções.

Um código predefinido já está presente no arquivo [main.js](#):

```
function preload() {  
  world_start = loadSound("world_start.wav");  
  setSprites();  
  MarioAnimation();  
}  
  
function setup() {  
  canvas = createCanvas(1240,336);  
  instializeInSetup(mario);  
}  
  
function draw() {  
  game()  
}
```

Desse código, observaremos com mais atenção a função `setup()`:

```
function setup() {  
  canvas = createCanvas(1240,336);  
  instializeInSetup(mario);  
}
```

1. O código para ler o canvas já lhe foi fornecido.
- 2.

P: A partir desse código de criação do canvas, você pode dizer qual é a altura e largura dele?

R: *Width* é 1240 e *Height* é 336.

Não modificaremos essas dimensões, pois são específicas para o jogo.

3. A segunda linha é `instializeInSetup(mario);`. Ela organizará todas as variáveis e funções necessárias para o jogo. Não precisamos nos aprofundar nisso agora, pois não estamos preocupados com como o jogo inicia mas com como ele é controlado.

Você se lembra da div que criamos na última aula para conter o canvas? Agora, escreveremos o código para colocá-lo dentro dela.

`parent()` é uma função p5.js que utilizamos para colocar o canvas ou qualquer outro componente p5.js, como a visualização da webcam, em uma div HTML.

Sintaxe: **variavelComComponente ou componenteP5.parent("ID HTML")**:

Aqui, queremos colocar o canvas dentro da div HTML, então o código será:

```
function setup() {  
  canvas = createCanvas(1240,336);  
  canvas.parent('canvas');  
  instializeInSetup(mario);  
}
```

- A variável canvas contém o canvas p5.js, por isso `canvas.`
- Em seguida a função p5.js `canvas.parent(`
- Depois a ID HTML da div que criamos para conter o canvas `canvas.parent('canvas');`

Como pode ver, a div HTML, que definimos para conter o canvas, está dentro da tag center.

OBSERVAÇÃO: A imagem abaixo está presente na **Atividade da Professora 2** para representação visual. Por favor, abra a **Atividade da Professora 2** enquanto explica.

```
<body background="background.jpg">
  <center>
    <div class="btn btn-primary heading">
      <h3>IA MARIO</h3>
      <button class="btn btn-primary" data-toggle="modal" data-target="#myModal">Instruções</button>
    </div>
    
    <br><br>
    <button class="btn btn-success" onclick="startGame()" id="start">Jogar</button>
    <br><br>
    <h3 id="status" class="btn btn-warning"></h3>
    <br><br>
    <div id="canvas"></div>
    <div id="gameConsole"></div>
    <h4>
      Source Code: <a href="https://github.com/linuk">Linuk</a> || Game Credits: Mario
    </h4>

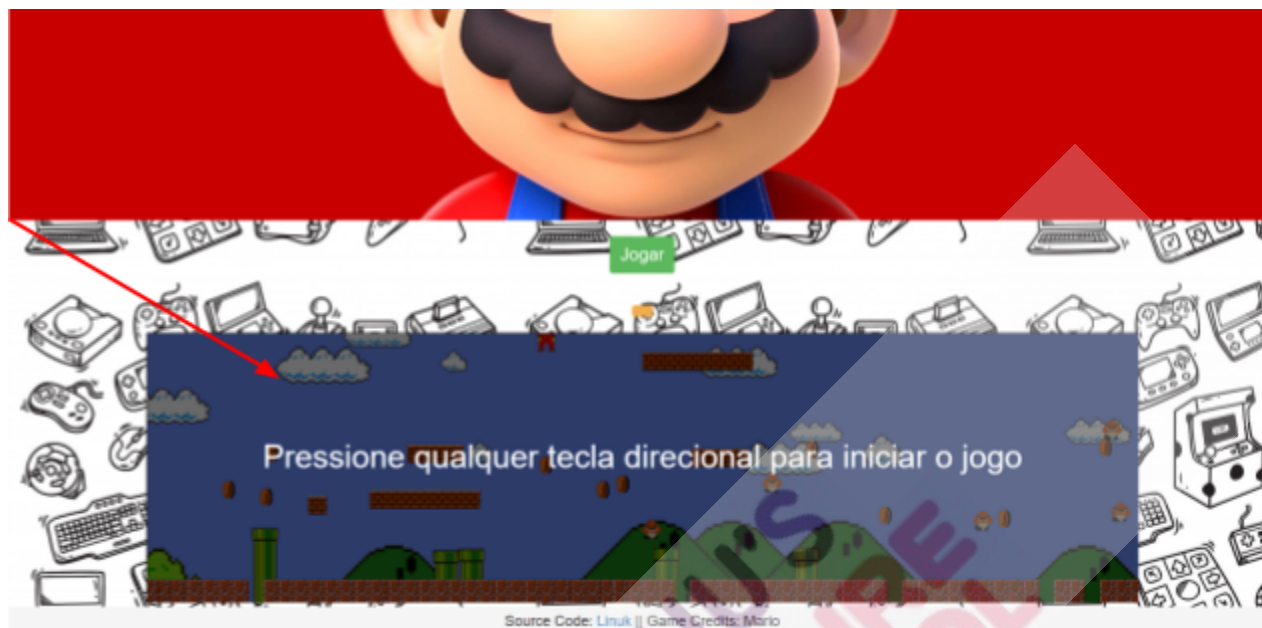
    <div id="myModal" class="modal fade ">
      <div class="modal-dialog">
        <div class="modal-content">
          <div class="modal-header">
            <button class="close" data-dismiss="modal">&times;</button>
            <h4>Instruções</h4>
          </div>

          <div class="modal-body">
            
            
            
          </div>
        </div>
      </div>
    </div>
  </center>
</body>
```

Qualquer elemento dentro da tag **center** será exibido no centro da página. Isso significa que essa **div** (definida para conter o canvas) será posicionada no centro da página web.

E, como armazenamos o canvas p5.js **nessa div**, ele também será colocado no centro do website.

Saída:



OBSERVAÇÃO:

Sempre que o aluno for executar o código, **CERTIFIQUE-SE DE QUE ELE FAÇA ISSO CLICANDO NO BOTÃO GO LIVE DO VISUAL STUDIO. COM ISSO, OS ARQUIVOS SERÃO EXECUTADOS NO LIVE SERVER DO VISUAL STUDIO**



Visto que utilizamos arquivos de áudio e imagem, e a biblioteca p5.js não nos permite executar esses arquivos no sistema local, precisaremos executá-los em um **server**.

A professora encerra o compartilhamento de tela

Agora é sua vez.

- Peça para o aluno pressionar a tecla ESC para retornar ao painel
- Ajude o aluno a iniciar o compartilhamento de tela
- A professora ativa o modo tela cheia

Passo 3:
Atividade Dirigida
pelo Aluno
(10 min)

Você deve continuar o código HTML no arquivo [index.html](#) e o código CSS em [style.css](#).

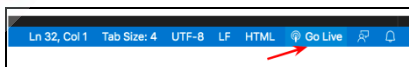
Hospedaremos o Jogo Mario com IA apenas quando o código estiver completo. Portanto, mantenha seus arquivos em segurança e, por favor, não os modifique, pois, como eu disse antes, qualquer alteração pode corromper o jogo.

Atividade do Aluno 1 - DIAGRAMA DO CÓDIGO

O aluno deve continuar programando em HTML no arquivo [index.html](#) e em CSS no arquivo [style.css](#).

Após o aluno completar o código, segundo as instruções, peça para que **execute o código**.

CERTIFIQUE-SE DE QUE O ALUNO REALIZE OS TESTES CLICANDO NO BOTÃO GO LIVE DO VISUAL STUDIO. COM ISSO, O CÓDIGO SERÁ EXECUTADO NO LIVE SERVER DO VISUAL STUDIO



Visto que utilizamos arquivos de áudio e imagem, e a biblioteca p5.js não nos permite executar qualquer arquivo de áudio ou imagem do sistema local, precisaremos executá-los em um **server**.

Hospedaremos o Jogo Mario com IA apenas quando o código para o jogo estiver completo.

A professora ajuda o aluno a encerrar o compartilhamento de tela

Passo 4:
Encerramento
(3 min)

Você se saiu muito bem hoje.
Ótimo! Você receberá duas tiradas de chapéu.

P: Adicionar a class **modal-dialog** à div modal fornecerá qual estilo?

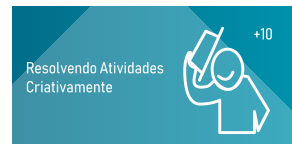
R: Essa class fornecerá *padding* ao pop-up, para que ele não toque nas extremidades da tela.

P: Adicionar a class **modal-content** à div modal fornecerá qual estilo?

R: Essa class fornecerá a cor branca de fundo, cor de fonte preta, *box-shadow* e *border-radius* ao pop-up.

(Dê, no mínimo, 2 tiradas de chapéu)

Pressione o ícone de Tirada de Chapéu para **Resolvendo Atividades Criativamente**.



Pressione o ícone de Tirada de Chapéu para **Ótima Pergunta**.



Pressione o ícone de Tirada de Chapéu para **“Você se Concentrou”**.



Caso não haja tempo para realizar as Atividades Adicionais, peça para o aluno realizá-las após a aula. As Atividades Adicionais estão presentes apenas no painel sob Atividades do Aluno

Além disso, lembre o aluno de consultar as Atividades de Referência do Aluno para ampliar seu conhecimento. Isso também deve ser feito após a aula.

Para a solução de todas as Atividades Adicionais, abra a **Atividade da Professora 4** e navegue até a aula **C139**

Atividade Adicional 1:

Execute a **Atividade do Aluno 2** do **painel**

A **ATIVIDADE** e as **DICAS** são mencionadas no próprio website.

Atividade Adicional 2:

Execute a **Atividade do Aluno 3** do **painel**

A **ATIVIDADE** e as **DICAS** são mencionadas no próprio website.

Atividade Adicional 3:

Execute a **Atividade do Aluno 4** do **painel**

A **ATIVIDADE** e as **DICAS** são mencionadas no próprio website.

Atividade Adicional 4:

Execute a **Atividade do Aluno 5** do **painel**

A **ATIVIDADE** e as **DICAS** são mencionadas no próprio website.

Atividade Adicional 5:

Execute a **Atividade do Aluno 6** do **painel**

A **ATIVIDADE** e as **DICAS** são mencionadas no próprio website.

A professora clica em

× Terminar Aula

Atividade	Nome da Atividade	Links
Atividade da Professora 1	JOGO MARIO IA	https://amadolucas.github.io/IA_MARIO/
Atividade da Professora 2	DIAGRAMA DO CÓDIGO	https://s3-whjr-curriculum-uploads.whjr.online/ecf637b4-6b33-41f9-acbf-b6e6e30b4bc6.pdf
Atividade da Professora 3	CÓDIGO FONTE COMPLETO	https://drive.google.com/file/d/1pPPo-x01UXrB3NtZenU_9Xj3UxZVqxSa/view?usp=sharing
Atividade da Professora 4	SOLUÇÃO DAS ATIVIDADES ADICIONAIS	https://docs.google.com/spreadsheets/d/e/2PACX-1vSG0V0IQyGx7Zo7VJgaWAHYDpalldCZ-RepABc_slelct65DaM-R9YF0vCbS5JFdsF3MEFQ-WLslsRc/pubhtml
Atividade do Aluno 1	DIAGRAMA DO CÓDIGO	https://s3-whjr-curriculum-uploads.whjr.online/ecf637b4-6b33-41f9-acbf-b6e6e30b4bc6.pdf
Atividade de Referência do Aluno1	REFERÊNCIA BOOTSTRAP MODAL	https://www.w3schools.com/bootstrap/bootstrap_modal.asp
Atividade de Referência do Aluno 2	REFERÊNCIA FUNÇÃO p5.js PARENT()	https://p5js.org/reference/#/p5.Element/parent
Solução do Projeto	IA PING PONG - PARTE 2	https://s3-whjr-curriculum-uploads.whjr.online/969f4bf1-75ce-4d5f-ab61-004c5dfceb3c.pdf