

Inputs de mouse

Transcrição

Nossa nave já se move, porém apenas em uma direção. Nosso próximo passo é fazer com que ela siga a direção do mouse. Como comentamos antes, um objeto na cena pode ter vários componentes associados ao mesmo. Clicando na nave, por exemplo, e verificando a aba **Properties** do objeto, teremos o componente **node** que não apenas indica o posicionamento do objeto, mas também dá informações de direcionamento.



A ideia é que capturemos o evento do próprio objeto agora e não da Cocos no geral. É o componente **node** quem vai nos auxiliar neste caso, nele podemos usar o método **on** que é o mesmo método da **SystemEvent** da Cocos, porém, direcionada ao objeto específico. Assim, no **onLoad** do **Jogador.js** teremos mais um registro de captura de evento.

```
// use this for initialization
onLoad: function () {
    cc.SystemEvent.on(cc.SystemEvent.EventType.KEY_DOWN, this.teclaPressionada, this);
    cc.SystemEvent.on(cc.SystemEvent.EventType.KEY_UP, this.teclaSolta, this);

    this.node.on("mousemove", this.mudarDirecao, this);
},
```

A Cocos não disponibiliza constantes para captura do movimento do mouse, mas podemos passar uma *string* com o nome do evento a ser capturado. Lembre-se, estamos executando nosso jogo no navegador, então podemos usar de seus recursos para algumas coisas. O evento é o **mousemove** e o método que criaremos para lidar com a direção da nave, considerando o mouse, é o **mudarDirecao**.

```
mudarDirecao: function(event){
    console.log(event);
}
```

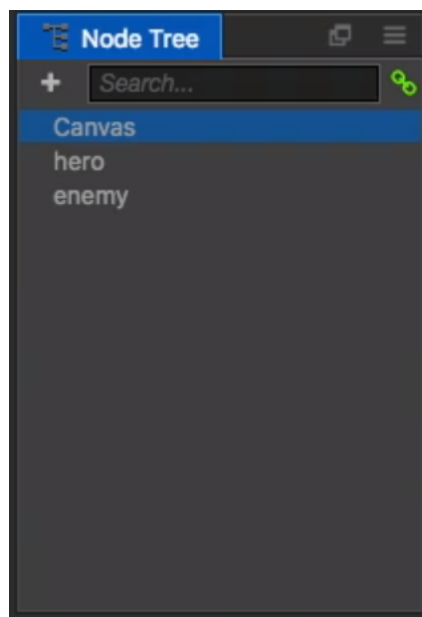
```
}
```

Por enquanto, vamos apenas imprimir o evento no console do navegador para testes. Note que ao executar o jogo nada acontece ao movermos o mouse na tela. Lembre-se de abrir a console do navegador para verificar se as mensagens de movimento do mouse estão sendo impressas. Nem mesmo estas mensagens estão sendo impressas, por quê?

Acontece que nós associamos a captura do evento do mouse ao objeto diretamente. Isso quer dizer que o movimento do mouse será considerado apenas se ele acontecer sobre o objeto da nave. Não era isso que queríamos. O objetivo era que ao mover o mouse pela "câmera" do jogo, algo acontecesse e não sobre o jogador em si.

Organização da Node Tree

Comentamos anteriormente que o quadro roxo da cena era a câmera do jogo, mas na verdade, o nome real dessa área é o `Canvas`. Isso é fácil de verificar olhando a aba `Node Tree` da Cocos.



Acontece que temos uma árvore (*Node Tree*), onde temos um `Canvas` e outros objetos da cena. O nosso evento de mouse não deve considerar o objeto em si, mas sim o `Canvas`, que é a área visível do jogo.

Buscando objetos da cena

Uma vez que já sabemos o problema, podemos solucioná-lo. A solução é buscar o `Canvas` que está em nossa cena e associar o evento de `mousemove` a ele. Para isso a Cocos mais uma vez nos ajuda com o método `find`, onde passamos o nome do objeto e ele nos retorna o componente `node` do objeto. Assim teremos:

```
// use this for initialization
onLoad: function () {
    cc.SystemEvent.on(cc.SystemEvent.EventType.KEY_DOWN, this.teclaPressionada, this);
    cc.SystemEvent.on(cc.SystemEvent.EventType.KEY_UP, this.teclaSolta, this);

    let canvas = cc.find("Canvas");
    canvas.on("mousemove", this.mudarDirecao, this);
},
```

Com isso temos nosso primeiro passo para a movimentação da nave considerando o mouse pronto. O próximo é fazer com que a direção do mouse seja considerada no movimento.