Aula 10

ALÉM DE JOGAR,
EU FAÇO JOGOS!

Instituto COOPERFORTE
20 anos transformando realidades

cpdi
Comitê para Democratização da Informática

Material desenvolvido pela professora Juliana Oliveira para o projeto além de jogar eu faço jogos. Divulgação proibida.

O que vamos aprender hoje?

- Biblioteca Composer | Cenas

Biblioteca de Composer

A biblioteca Composer é o sistema oficial de criação e gerenciamento de cenas no Corona. Essa biblioteca fornece aos programadores uma maneira fácil de criar e fazer a transição entre cenas individuais.

O ciclo do Composer inicia no arquivo main.lua, porém esse arquivo não é uma cena do Composer. Ela é utilizada apenas para o código de inicialização, e em seguida, inicia a primeira cena pelo comando composer.gotoScene(). Nessa chamada, especificamos o arquivo com o nome da cena a ser carregada, sem a extensão.

Biblioteca de Composer

As cenas são basicamente scripts, porém, existem algumas configurações estruturais adicionadas que devemos obedecer para que o Composer os tratem propriamente como cenas. Isso vai incluir duas linhas para inicializar a cena, quatro funções de ouvinte para manipular os eventos que o composer gera, quatro linhas de código para inicializar essas funções de ouvinte e um return que associa a cena ao Composer. O objeto de cena é criado chamando o comando composer.newScene(). Esse objeto contém dados importantes para a cena que o Composer deve acessar. Para o programador, o aspecto mais importante é o self.view - grupo de exibição da cena, que é o grupo ao qual todo o conteúdo visual da cena deve ser adicionado.

Fluxo de cenas

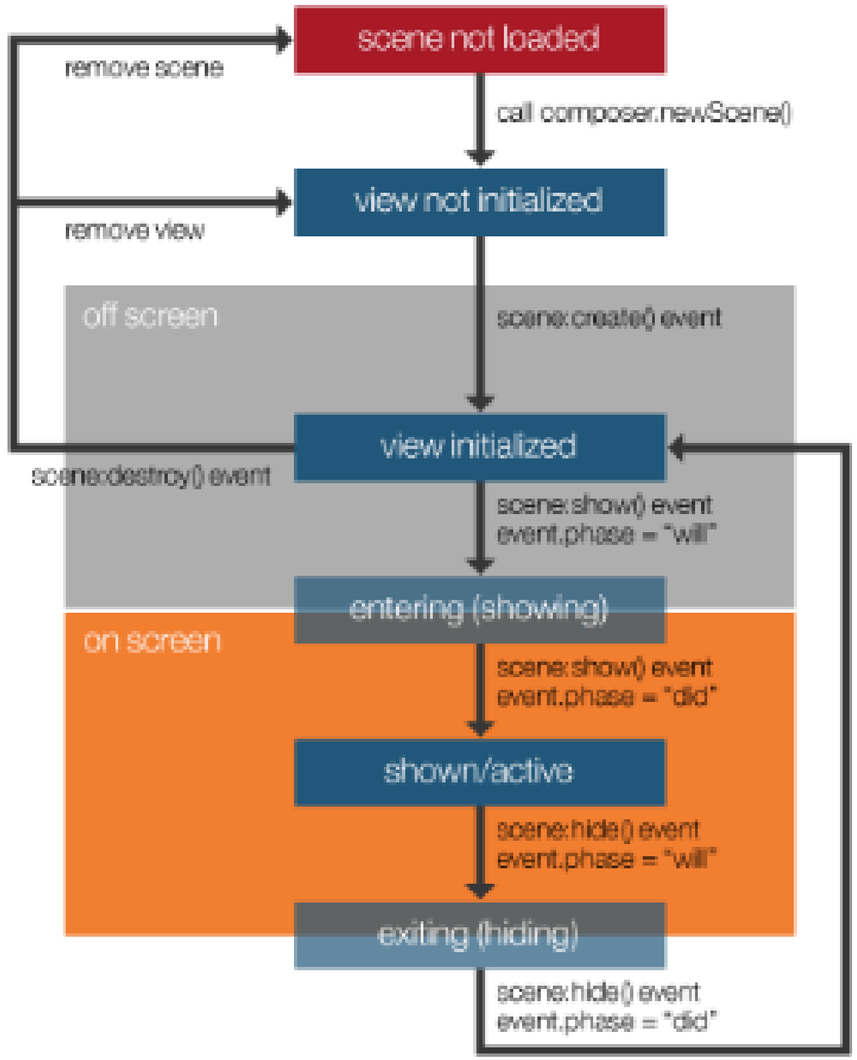
Quatro funções de ciclo de vida diferentes lidam com eventos gerados pelo Composer:

Função	Ponto de Ciclo de Vida
`scene:create()`	Ocorre quando a cena é criada pela primeira vez, mas ainda não apareceu na tela.
`scene:show()`	Ocorre imediatamente antes e/ou imediatamente após a cena aparecer na tela.
`scene:hide()`	Ocorre imediatamente antes e/ou imediatamente após a cena sair da tela.
`scene:destroy()`	Ocorre quando a cena é destruída.

Ciclo de vida

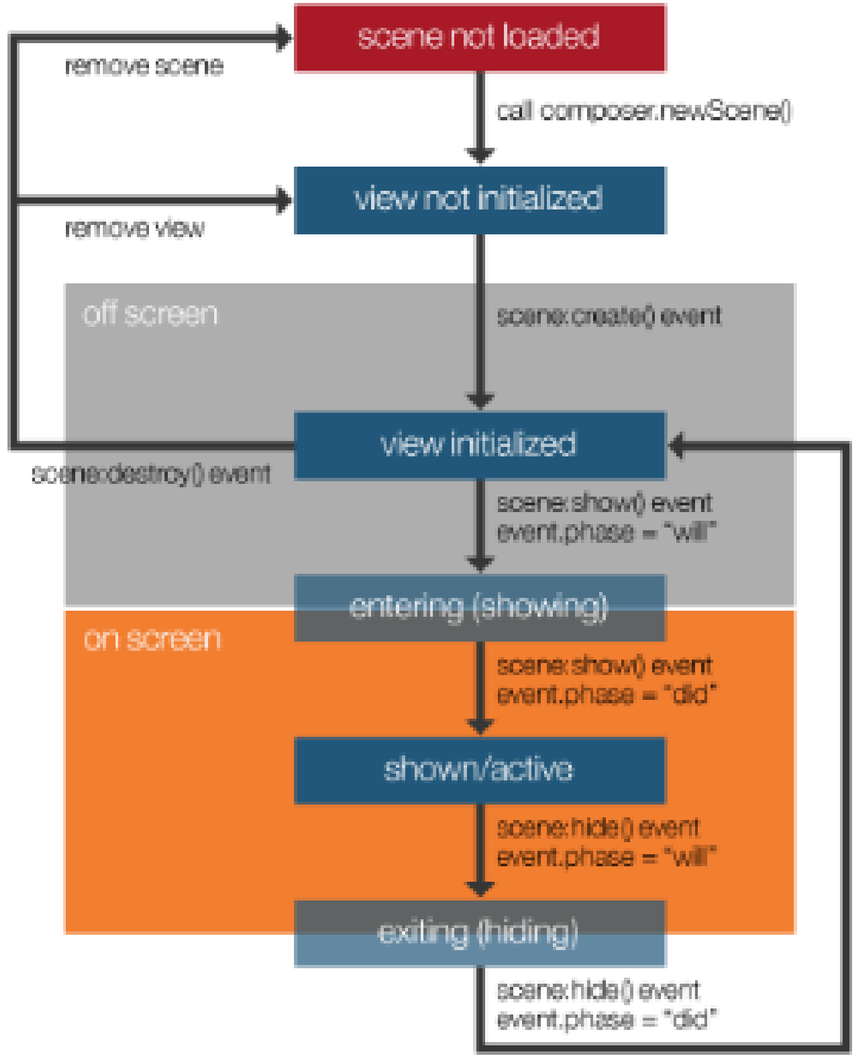
Uma vez que chamamos o comando gotoScene, o ciclo de vida da cena começa e segue um fluxo básico que é:

1. O arquivo .lua da cena é carregado e um objeto de cena Composer é criado pelo comando newScene. Nesse ponto, a visão da cena não é inicializada.



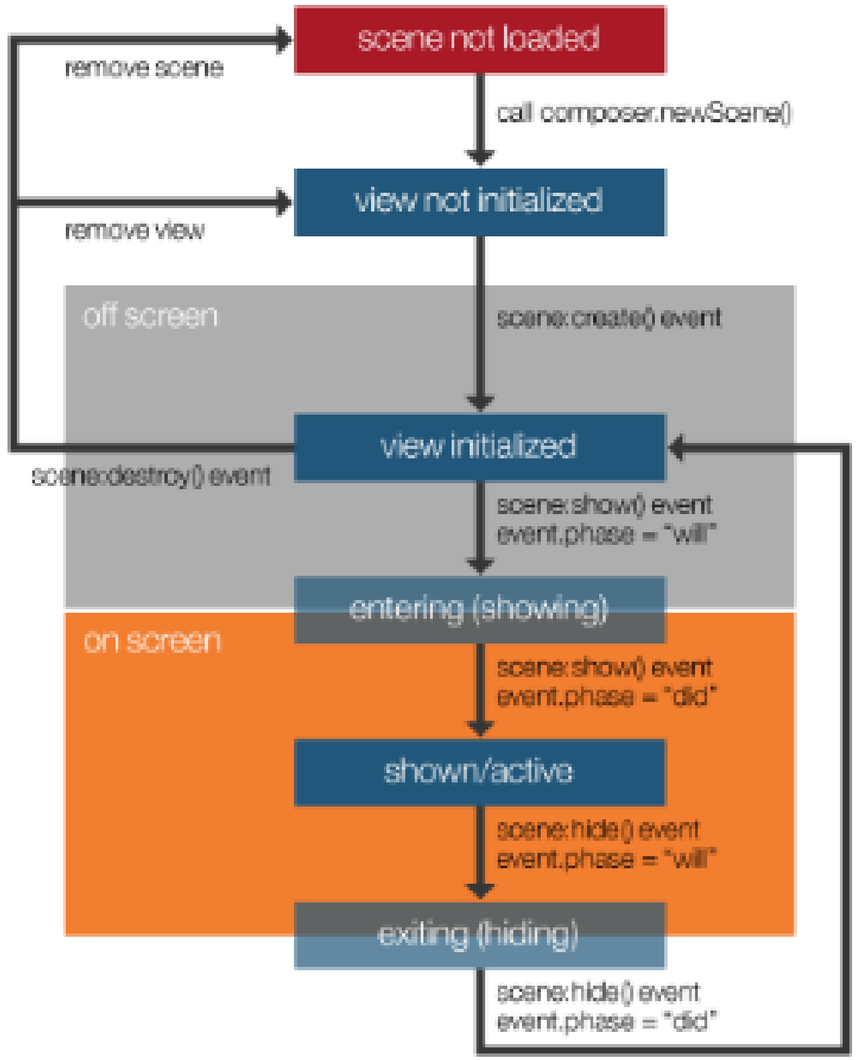
Ciclo de vida

2. A visão é inicializada e se ainda não existir, um evento de criação é despachado para a função da cena scene:create. Nesse ponto, a cena ainda reside fora da tela, então esse é o momento certo para criar objetos de interface do usuário e outros objetos de exibição necessários para a cena, incluindo botões, texto etc.



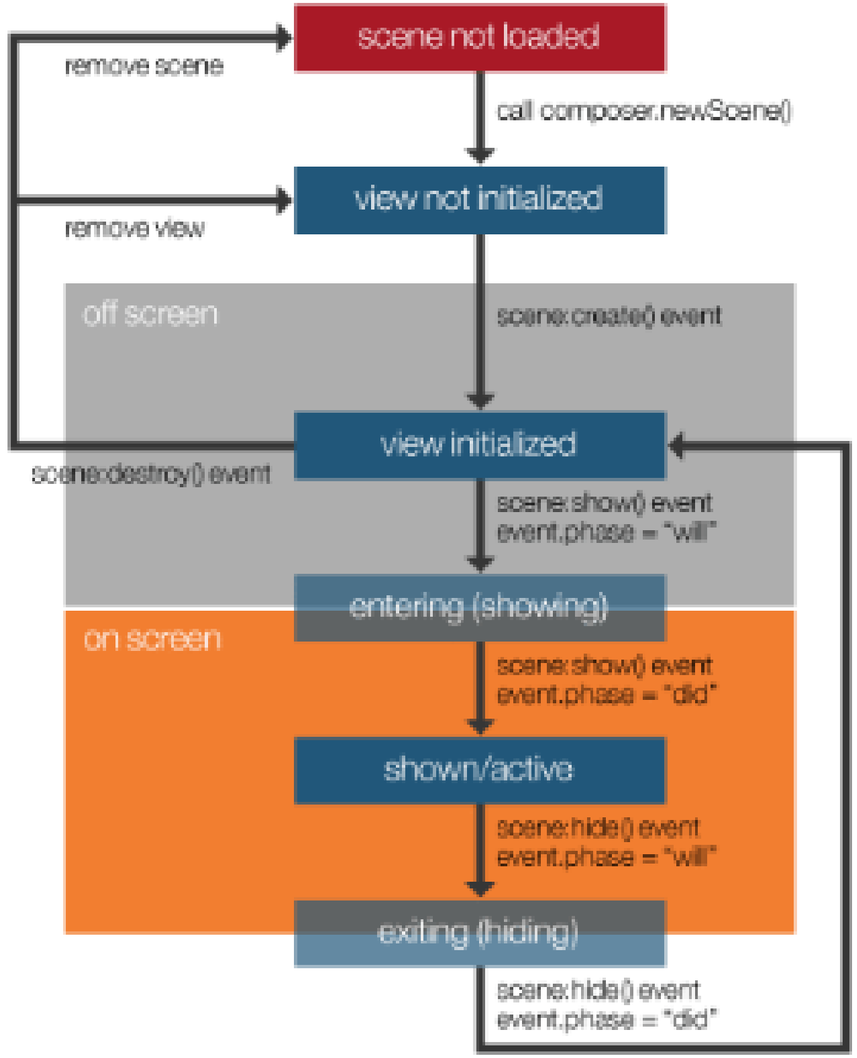
Ciclo de vida

3. Imediatamente antes da cena passar para a tela, um evento show é despachado para a função da cena scene:show com um parâmetro phase igual a "will" e essa é uma ótima oportunidade para redefinir valores de variáveis ou reposicionar objetos que podem ter se movido de sua posição inicial pretendida desde que a cena foi exibida da última vez. (por ex: ao reiniciar o nível do jogo).



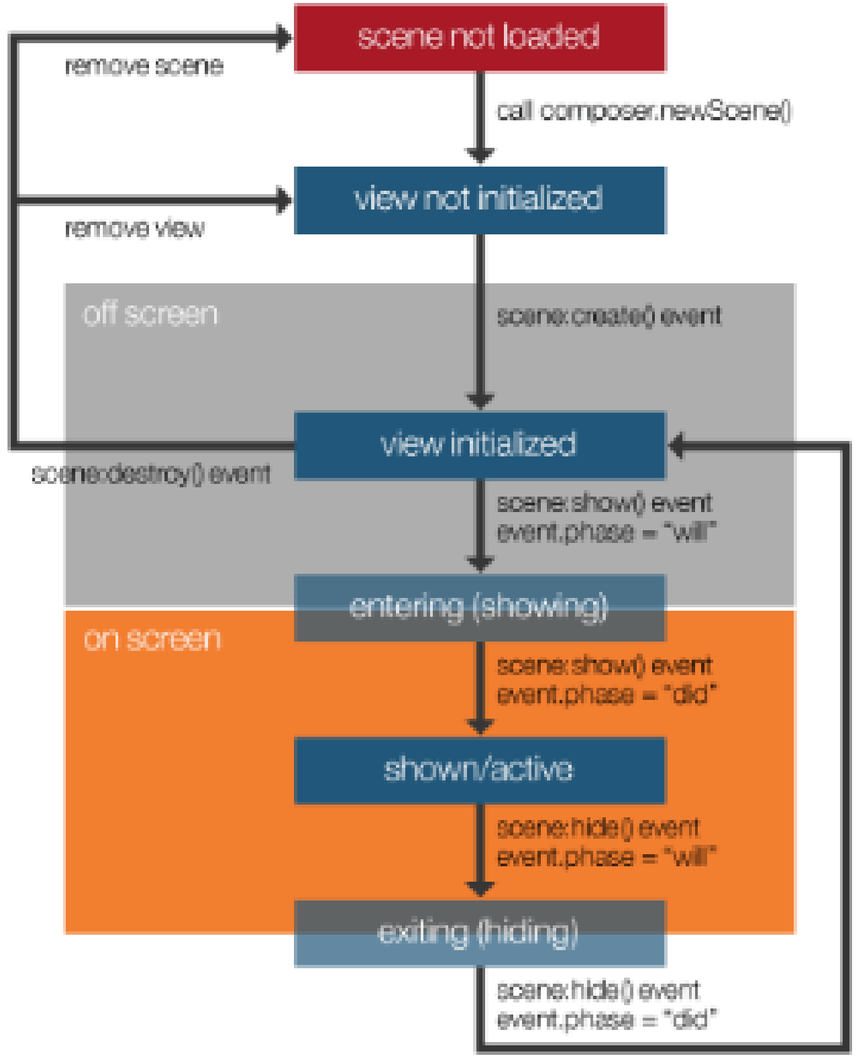
Ciclo de vida

4. Uma vez que a cena está totalmente na tela, outro evento show é despachado, porém com a fase de evento "did", essa cena agora é considerada a cena ativa. É um bom lugar para iniciar transições, tocar música de fundo específica da cena que foi carregada e iniciar a simulação de física se houver no projeto.



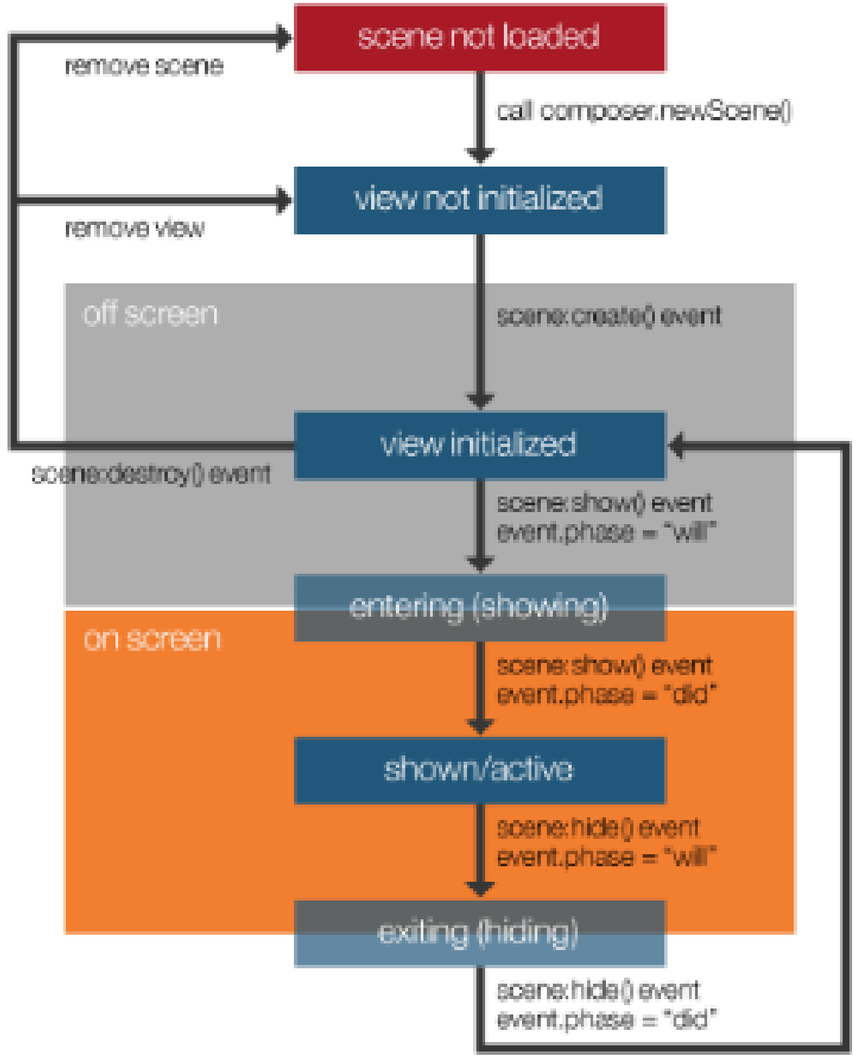
Ciclo de vida

5. Se sair dessa cena indo para outra por exemplo, um evento hide é despachado para a função scene:hide com um parâmetro de phase igual a "will". Oportunidade indicada para pausar ou parar a física, cancelar temporizadores e transições e parar o áudio específico da cena que foi reproduzido anteriormente.



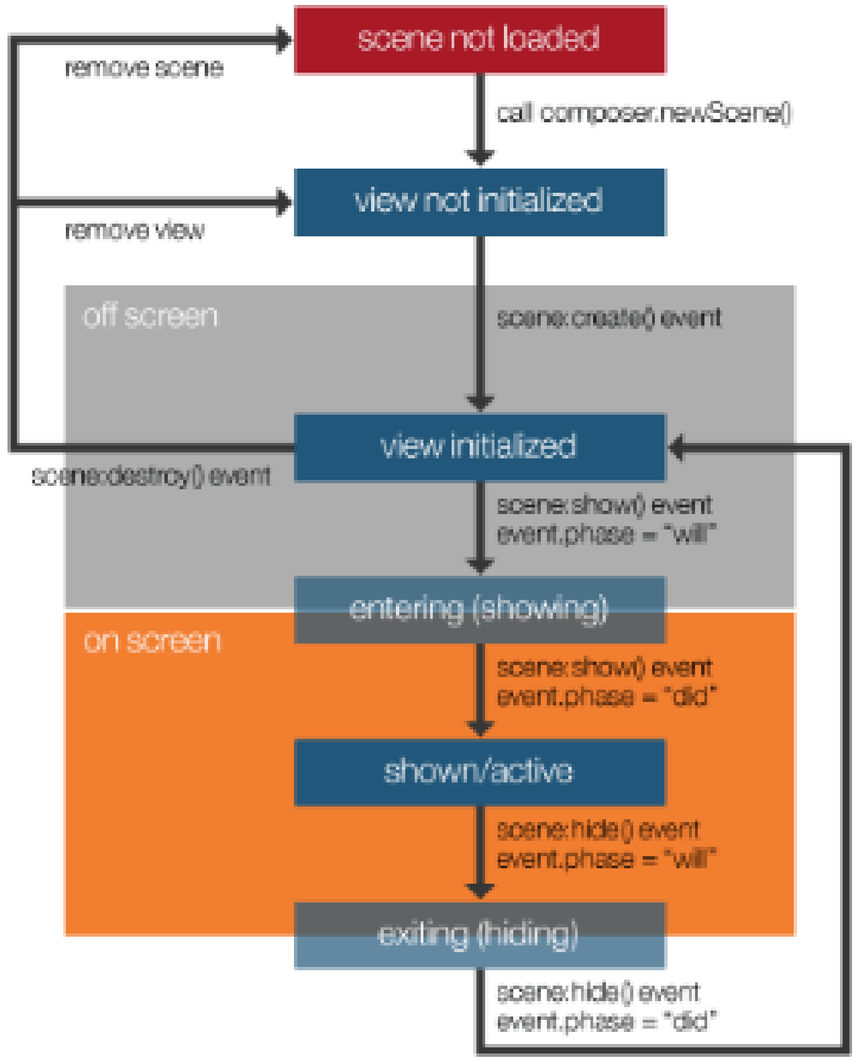
Ciclo de vida

6. Uma vez que a cena está totalmente fora da tela, outro evento hide é enviado com o parâmetro de phase igual a "did", nesse ponto, a visão da cena permanece inicializada pois o Composer mantém cenas ocultas/inativas na memória por padrão, supondo que possam ser exibidas novamente periodicamente.



Ciclo de vida

7. A partir desse ponto, se a cena for destruída, seja por comando ou como resultado de auto-reciclagem, um evento destroy é despachado para a função da cena scene:destroy. É quando o composer limpa os objetos de exibição da cena. Esse também é um bom momento para desfazer coisas feitas no create que não estão relacionadas aos objetos de exibição da cena, por exemplo descarte de áudio.



Indo para cenas

Depois que criar os arquivos de cena, precisamos de um método para acessá-los. No Composer, fazemos isso via composer.gotoScene()

Além disso, existem vários parâmetros para os quais podemos passar opcionalmente ao gotoScene para controlar a transição, ou fornecer dados para a cena. Podemos por exemplo:

- Definir o efeito de transição
- Definir o tempo de transição
- Passar uma tabela de parâmetros/dados para a cena.

Efeitos de transição

Temos vários efeitos compatíveis com o parâmetro effect para as transições de cenas.

- `"fade"`
- `"crossFade"`
- `"zoomOutIn"`
- `"zoomOutInFade"`
- `"zoomInOut"`
- `"zoomInOutFade"`
- `"flip"`
- `"flipFadeOutIn"`
- `"zoomOutInRotate"`
- `"zoomOutInFadeRotate"`
- `"zoomInOutRotate"`
- `"zoomInOutFadeRotate"`
- `"fromRight"` — sobre a cena atual
- `"fromLeft"` — sobre a cena atual
- `"fromTop"` — sobre a cena atual
- `"fromBottom"` — sobre a cena atual
- `"slideLeft"` — empurra a cena atual para fora
- `"slideRight"` — empurra a cena atual para fora
- `"slideDown"` — empurra a cena atual para fora
- `"slideUp"` — empurra a cena atual para fora

Gerenciamento de cena

Um recurso importante do Composer é que na maioria das vezes, ele gerencia os objetos de exibição automaticamente, supondo que sejam incluídos no grupo de visualização da cena. Após a cena ser reciclada ou removida, o Composer tratará do descarte adequado dos objetos que forem inseridos no grupo de visualização da cena, ou em grupos que estejam dentro do grupo de visualização. Ouvintes de tap, touch e colisão aplicados a esses objetos também serão removidos.

Assim como remover objetos de exibição fora do Composer, ainda somos responsáveis pelo seguinte quando terminar uma cena:

- Remover os ouvintes de execução;
- Cancelar transições e temporizadores;
- Descartar áudios carregados;
- Fechar todos os arquivos que foram abertos, incluindo banco de dados.

Cenas de reciclagem automática

Como citado anteriormente, o Composer mantém a visualização da cena atual na memória, o que pode melhorar o desempenho se acessarmos as mesmas cenas com frequência. Se desejarmos reciclar a cena ao mudar para uma nova, podemos definir a propriedade recycleOnSceneChange como true.

Para reverter para o comportamento padrão, é só definir a propriedade como false.

Vale lembrar que nenhuma dessas condições descarregará a cena da memória. Para remover explicitamente uma cena da memória, usamos o comando removeScene.

Removendo cenas

Quando terminamos completamente com as cenas e não pretendemos acessá-las novamente, podemos chamar duas funções:

- composer.removeScene() - Remove ou recicla uma cena específica.
- composer.removeHidden() - Remove ou recicla todas as cenas, exceto a que estiver ativa no momento.

Recarregando cenas

Recarregar cenas requer uma abordagem ligeiramente diferente. Muitas pessoas querem criar e posicionar todos os objetos de exibição de cena na função create. No entanto, se recarregarmos a cena a partir de si mesma, essa função não será chamada novamente pois a visão da cena ainda existe. Objetos que podem ter sido movidos permanecerão no lugar quando recarregar a cena, e as variáveis definidas fora das funções de ouvinte permanecerão em seus valores atuais.

Cenas de sobreposição

O Composer permite que tenhamos uma cena de sobreposição. É uma cena que é carregada no topo da cena ativa. Uma cena de sobreposição é construída como qualquer outra.

Mostrando uma sobreposição

Para mostrar uma sobreposição, chamamos pela função showOverlay. Como uma cena de sobreposição pode não cobrir toda a tela, o usuário pode interagir potencialmente com a cena que estiver abaixo. Para evitar isso, definimos o parâmetro isModal como true na tabela de opções. Isso faz com que os eventos de touch/tap não passem para a cena de baixo.

Escondendo uma sobreposição

Para ocultar uma sobreposição e retornar a cena pai, usamos o comando hideOverlay.

Ele pode ser chamado da cena de sobreposição, da cena pai ou de algum manipulador de eventos como um de teclas "back" do Android. A tentativa de ir para outra cena pelo comando gotoScene também ocultará a sobreposição automaticamente.

Acessando a cena pai

Ao mostrar ou ocultar a sobreposição, pode ser necessário realizar ações na cena pai. Por exemplo, podemos precisar reunir alguma entrada ou seleção da sobreposição e atualizar algum aspecto da cena pai. No Composer, a cena de sobreposição tem acesso ao objeto de cena do pai via event.parent. Isso permite acesso a funções/métodos na cena pai e se comunicar com o pai quando a cena de sobreposição for mostrada ou ocultada.