

**CONSTRUIR PACMAN**

Aprender moderna Javascript, HTML5 Canvas, y un poco de EmberJS

Jeffrey Biles

Este libro está a la venta en <http://leanpub.com/buildpacman>

Esta versión se publicó el 02/05/2016



Esto es un [Leanpub](http://leanpub.com) libro. Leanpub permite a los autores y editores con el proceso de publicación de Lean.[editorial magra](http://leanpub.com/manifesto) es el acto de publicar un libro electrónico en curso utilizando herramientas ligeras y muchas iteraciones para obtener retroalimentación de los lectores, de pivote hasta que tenga el libro adecuado y construir la tracción vez que lo hace.

© 2015 - 2016 Jeffrey Biles

**Contenido**

|  |  |
| --- | --- |
| **En Media Res** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **1** |
| **1: Dibujo Un círculo** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **3** |
| Instalación. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 3 |
| La creación de su aplicación. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 4 |
| Viendo una caja. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 4 |
| Dibujar un círculo. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 5 |
| Instalación completa . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 6 |
| **Formación** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **7** |
| **2: ponerse al día,** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **8** |
| Componentes. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 8 |
| Manillar. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 9 |
| Lienzo. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 9 |
| Css. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 9 |
| contexto de la lona y dibujo. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 10 |
| Dejar . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 10 |
| didInsertElement. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 11 |
| Funciones. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 11 |
| importación y exportación . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 13 |
| El sentirse bien todavía? . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 14 |
| **Controlar** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **15** |
| **3: Entrada** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **dieciséis** |
| Robar este código. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | dieciséis |
| Llevándolo en. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | dieciséis |
| Mixins. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 17 |
| La definición de atajos de teclado. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 17 |
| Hashes. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 18 |
| Tala. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 19 |
| Resumen . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 20 |
| **Desplazamiento** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **21** |

CONTENIDO

1. **Movimiento** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 23

|  |  |
| --- | --- |
| Propiedades. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 23 |
| Actualización de Propiedades. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 24 |
| Traslado de la PAC. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 25 |
| Borrado de la pantalla. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 26 |
| La extracción de métodos. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 27 |
| Una propiedad calculada. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 28 |
| Resumen . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 29 |
| **barreras** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **30** |

1. **Recinto** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 32

|  |  |
| --- | --- |
| Alineación. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 32 |
| Si los estados. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 34 |
| colisiones completamente funcional. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 36 |
| Encuadernación manillar. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 36 |
| Resumen . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 38 |
| **Dentro del laberinto** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **39** |

1. **Las barreras internas** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 41 Dibujo de una pared. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 41

|  |  |
| --- | --- |
| Paredes, dispuestas ante nosotros. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 41 |
| Dibujo todas las paredes. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 42 |
| Llamar al sorteo. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 43 |
| Las colisiones pared. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 43 |
| Resumen . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 45 |
| **Despejando el camino** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **46** |

1. **La cuadrícula** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 48 Al ver la cuadrícula. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 48

|  |  |
| --- | --- |
| Matrices de matrices. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 49 |
| Viendo la red. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 49 |
| El cálculo de la anchura y la altura de nivel. . . . . . . . . . . . . . . . . . . . . . . . . . . . | 51 |
| Profunda se. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 52 |
| Mueve a través del espacio. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 53 |
| Resumen . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 54 |
| **primera cosecha** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **56** |

1. **Los pellets** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 58

|  |  |
| --- | --- |
| Dibujo Los pellets. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 58 |
| Di lo que haces, haz lo que dices. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 60 |
| Los problemas de alcance. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 61 |

CONTENIDO

Las flechas de grasa. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 62 Comparte tu círculo. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 63 El procesamiento de los pellets. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 65 Resumen. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 67

**Ganancias y pérdidas** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . **68**

1. **Puntajes y niveles** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 69

|  |  |
| --- | --- |
| Manteniendo el marcador . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 69 |
| Elevar a mismo nivel . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 70 |
| Nueva Sintaxis. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 72 |
| Resumen . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 73 |
| **Suave, eficiente** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **74** |

1. **animaciones** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 76 Direcciones de coordinación. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 76

|  |  |
| --- | --- |
| nextCoordinate y Cuerdas interpolada. . . . . . . . . . . . . . . . . . . . . . . . . . . | 77 |
| Mirando hacia el futuro con la red. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 78 |
| Las animaciones suaves. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 79 |
| Guardando nuestro movimiento. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 80 |
| El Loop Movimiento. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 81 |
| Ember.run.later. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 82 |
| Dibujo con FrameCycles. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 82 |
| Resumen . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 84 |
| **ninguna detención** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **85** |

1. **El bucle del juego** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 87

|  |  |
| --- | --- |
| Rollin Rollin Rollin. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 87 |
| El sistema de Intención. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 88 |
| Resumen . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 90 |
| **objetivar** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **91** |

1. **Separación de intereses** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 93 Definición de nuevos archivos. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 93

|  |  |
| --- | --- |
| El retorno de los módulos ES2015. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 93 |
| Sistemas con clase. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 94 |
| Mixins. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 95 |
| Las clases e instancias. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 96 |
| Resumen . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 97 |
| **lealtades** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **98** |

CONTENIDO

1. **Separación de intereses** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 100

|  |  |
| --- | --- |
| Separación: Pac objeto. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 100 |
| Separación: SharedStuff Mixin. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 102 |
| Separación: pac-man componentes. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 103 |
| Separación: extracción parcial método. . . . . . . . . . . . . . . . . . . . . . . . . . . . | 106 |
| Resyncing en Reiniciar. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 108 |
| Resumen . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 109 |
| **Separación** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **110** |

1. **Nivelarse** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 112

|  |  |
| --- | --- |
| El objeto de nivel. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 112 |
| Reiniciar todo. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 114 |
| Hacer que todo parezca. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 115 |
| LessSharedStuff. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 116 |
| Otro nivel . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 117 |
| El Nivel gana más poder regulador. . . . . . . . . . . . . . . . . . . . . . . . . . . | 118 |
| Resumen . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 119 |
| **En primer lugar que frecuenta** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **120** |

1. **Fantasmal** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 122

|  |  |
| --- | --- |
| El fantasma de sí mismo. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 122 |
| Colores verdaderos . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 124 |
| Movimiento: Está vivo! . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 124 |
| El calor de búsqueda de fantasmas. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 127 |
| Mapa . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 128 |
| ¿Dónde está el Pac? . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 129 |
| El (Weighted Random) Chase. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 130 |
| Resumen . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 132 |
| **Sacrificio** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **133** |

1. **Contacto** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 134

|  |  |
| --- | --- |
| Varios fantasmas. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 134 |
| Mejor las funciones de flecha. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 136 |
| Detección de colisiones . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 136 |
| Lugares, por favor. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 138 |
| Spread super (opcional). . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 138 |
| Nivel sabe mejor. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 139 |
| Vidas extra. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 140 |
| Resumen . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 143 |
| **Restos** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **144** |

CONTENIDO

1. **Moving On Down** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 146

|  |  |
| --- | --- |
| La matriz de nivel. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 146 |
| Cambio de los niveles. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 147 |
| Resumen . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 149 |
| **Patrulla** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **151** |

1. **Teleport** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 152

|  |  |
| --- | --- |
| Encontrar lo que a cambio. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 152 |
| A dónde vamos desde aquí . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 153 |
| Encendiéndolo. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 154 |
| Un Teleport-Primer Nivel. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 154 |
| Resumen . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 156 |
| **Una nueva esperanza** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **157** |

1. **Poder** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 159

|  |  |
| --- | --- |
| En el mapa . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 159 |
| Modificación de los colores, operadores ternarios. . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 161 |
| El restablecimiento del nivel. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 162 |
| Resumen . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 163 |
| **El PAC atacan de nuevo** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **164** |

1. **Cambio de rumbo** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 165

|  |  |
| --- | --- |
| Santo Ejecutar. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 165 |
| Los operadores de la taquigrafía. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 166 |
| Las colisiones más específicas. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 166 |
| Retiro completo. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 168 |
| Reagruparse. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 168 |
| Resumen . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 170 |
| **El retorno del jihadista** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **171** |

1. **Se acabó el tiempo** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 172

|  |  |
| --- | --- |
| Temporizadores. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 172 |
| Menguante poder. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 173 |
| Un nuevo caso. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 173 |
| Modificación de los colores. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 174 |
| RGB. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 175 |
| La mezcla de colores. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 176 |
| Retirarse, hecha de nuevo. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 177 |
| Resumen . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 178 |
| **Epílogo** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | **180** |

CONTENIDO

**¿Que sigue?** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 181

**En Media Res**

**[Advertencias contenido: La religión, la violencia, Geopolítica, creativo Puntuacion. No dude en leer sólo los capítulos de instrucción si estas cosas son vas a mear fuera (o si estás en el trabajo)]**

Que se está ejecutando a través de la selva.

El por qué ahora parece distante, perdida en los laberintos oscuros muchas que ha hecho. Todo lo que sientes ahora es el látigo de ramas en su cara, el crujido de las ramas bajo los pies, y la chuleta chuleta chuleta de un helicóptero que se aproximaba.

El helicóptero de repente salta a la vista. Está cerca. Y hay un tipo sentado allí con una ametralladora. “Por fin te tengo!”, Grita. Él comienza a disparar, y los árboles se astilla y caer bajo el chorro de balas.

Uno de esos árboles cae sobre ti. Atrapado. Estás atrapado. El helicóptero se acerca. Quiere ver su rostro cuando se muere.

Pero entonces un cohete estalla en el helicóptero, y está en llamas. Su perseguidor salta lejos de lo que pronto se llama de escombros. Aterriza cerca de ti, tan cerca que se puede escuchar el crujido repugnante de sus huesos romperse. Su bolita de energía se había agotado.

Todavía se vuelve hacia ti, usando sus brazos para arrastrar su cuerpo destrozado más cerca. Su cara tensa por el esfuerzo, el dolor, el odio, pero en ella se ve algo familiar.

Se derrumba, sin aliento, y se hace más clara.

Mi dios, es él.

Es él.

Él te mira y ve la misma cosa. Los desagües rabia de la cara. Ahora ... ahora es sólo dolor.

Sólo la memoria.

“Nunca voy a olvidar”, dice. “Nunca olvidaré las cuevas”.

Y de repente estás de vuelta allí con él. Esas cuevas oscuras, esas habitaciones ennegrecidas, con sólo los pellets y la persecución.

El miedo constante.

Son abstracciones ahora, pero en aquel entonces eran muy reales.

“No sé por qué se volvió,” dice. “Creo que nunca lo sabré.”

Se empieza a explicar, empezar a poner excusas, pero las palabras no vienen. Un hombre inexperto estaría llorando, pero no están capacitados. En su lugar, sólo se dice: “Usted será bendecido en el cielo por lo que ha hecho.”

1

|  |  |
| --- | --- |
| En Media Res | 2 |

“¿Es demasiado tarde para usted?”, Se pregunta. "¿Puedo verte ahi?"

Una bala y que se ha ido. El árbol despega de ti, y se extiende una mano, tirando para arriba.

Usted debe sentirse aliviado.

Estás seguro.

Sus compañeros de fantasmas han llegado.

**1: Dibujo Un círculo**

Este capítulo es muy parecida a la de uno anterior extraño, lleno de personajes que no entienda, y finalmente volviendo torno a Pac-Man. La única diferencia es que en este capítulo, vamos a hacer un montón de código.

Es una versión de In Media Res- una técnica de narración utilizado para llegar a la gente en la acción rápidamente. Se abre en el centro de la historia, durante una escena muy emocionante, y luego más tarde volver atrás y construir los personajes y las relaciones que hicieron posible esa escena.

Vamos a utilizar una técnica similar para este capítulo- que vamos a entrar de lleno en la instalación de un montón de herramientas y el uso de algunos conceptos no puede entender por el momento. El resultado es que se va a dejar a empezar dibujando círculos (pequeño prototipo PAC-hombres) en nuestro lienzo muy rápido. A partir de ahí podemos tomar las cosas con calma.

Una nota sobre las herramientas:

Vamos a utilizar una gran cantidad de herramientas en este libro-Javascript, Babel.js (una herramienta que nos permite usar las últimas versiones de Javascript antes de que estén en todos los navegadores), HTML5 Canvas, Ember.js, brasa-cli , y más. Originalmente, el libro iba a ser un libro para los profesionales de concentrarse sólo en ES2015 y HTML5 Canvas, pero he descubierto que el uso de más de estas herramientas en realidad hace que nuestro trabajo sea más fácil.

El objetivo de este libro no es ser una guía concluyentes en cualquiera de estas tecnologías, en que es para darle suficiente conocimiento de la herramienta para empezar, y luego usar esa herramienta para lograr nuestro objetivo (crear un juego malo-culo de la PAC -hombre). Sin embargo, al final del libro que estará familiarizado con varios conceptos- realmente útil y si decide seguir aprendiendo, voy a ofrecer varios recursos para cada uno.

**Instalación**

El primer paso es la instalación de nodo. Ir a https://nodejs.org/ y pulsa el botón verde grande “instalar”. Hacer lo que tu sistema operativo hace que hacer para instalar cosas, y entonces usted tiene nodo y NPM en su sistema. Vaya a la línea de comandos y escriba en-v nodo. A continuación, escribaNPM -v. Si los comandos trabajadas y que te dieron los números, hurra! Si no es así, por favor google en busca de ayuda. Este no es un libro sobre el nodo y NPM ... son sólo herramientas que necesitamos para empezar.

A continuación, vamos a instalar brasa-cli. TipoNPM instalar -g brasa-clien la línea de comandos. Y eso es. Tipo-v brasa en su línea de comandos, y debería darle la versión brasa-cli (> = 2.3.0) y las versiones de nodo y de la NGP.

3

|  |  |
| --- | --- |
| 1: Dibujo Un círculo | 4 |

**La creación de su aplicación**

brasa nueva pac-man

Este comando va a crear para usted una aplicación brasa llamado Pac-Man. Se va a pasar un corto periodo de tiempo la creación de una estructura de directorios y algunos archivos de configuración para usted, y luego un poco más la instalación de paquetes de la NGP y Bower.

cd pac-man

servidor brasa

Estos comandos le llevaría a la carpeta de su aplicación y luego iniciar el servidor. Ir a localhost: 4200 y debería ver “Bienvenido a Ember.js”. Si lo hace, entonces las felicitaciones! Que haya creado su primera aplicación brasa.

**Viendo una caja**

brasa generar componente pac-man

Escriba esto en la línea de comandos y obtendrá un nuevo componente llamado brasa pac-man. brasa-cli generará el archivo JavaScript, archivo de plantilla, y la prueba de un componente.

En app / templates / application.hbs, borrar todo, reemplace con el componente {{Pac-man}}.

Esto le dice brasa para mostrar el componente. A partir de ahora, todo lo que va a ocurrir en el interior del componente.

* *<! - en app / templates / componentes / PAC-man.hbs ->*
* <**lona** carné de identidad="myCanvas" anchura="800" altura="600"> </**lona**>

Esto crea un canvas de HTML5 que podemos sacar adelante.

* */ \* En app / estilos / app.css \* /*

2 **#myCanvas** {

3color de fondo: # EDC9AF;

4 }

Este colores del lienzo para que lo puedan ver. Es una luz de color marrón desierto de arena.

Si todo ha ido correctamente, en localhost: 4200 debería ver un rectángulo del color de la arena del desierto.

|  |  |
| --- | --- |
| 1: Dibujo Un círculo | 5 |



**Dibujar un círculo**

* *// en app / componentes / PAC-man.js*

2 **importar** Ember desde 'ascua';

3

4 **por defecto de exportación** Ember.Component.extend ({

5didInsertElement: **función**() {

* **esta**.drawCircle ();
* },

8

9drawCircle: **función**() {

1. **dejar** lona=documento.getElementById ("myCanvas");
2. **dejar** CTX=canvas.getContext ("2D");
3. **dejar** X= 50;
4. **dejar** y= 100;
5. **dejar** radio= 20;

15

1. ctx.fillStyle = '# 000';
2. ctx.beginPath ();
3. ctx.arc (x, y, radio, 0, Mates.PI \* 2, **falso**);
4. ctx.closePath ();
5. ctx.fill ();
6. },
7. });

didInsertElement ejecuta cada vez que el componente (el elemento) se carga y poner en la pantalla (Más sobre esto en el siguiente capítulo). En este caso, sólo estamos llamando a la función drawCircle, que se define justo debajo.

|  |  |
| --- | --- |
| 1: Dibujo Un círculo | 6 |

Nota para los desarrolladores avanzados Ember (todos los demás ignoran): usando didInsertElementen este caso, se dará un aviso de desaprobación, pero por ahora es la mejor opción. Utilizandoen esodará un error porque necesitamos el html que ya ha sido prestado. UtilizandodidRender te sumergirá en un bucle infinito.

En la función drawCircle, estamos agarrando el lienzo, diciéndole que vaya a dibujar en 2D, y luego hacer los encantamientos que le dicen lienzo para dibujar un círculo. Usted no tendrá que preocuparse por estos específica incantations- este libro casi siempre ocultar código de forma de dibujo detrás de una función que sólo se puede copiar y pegar sin perder mucho.



**Instalación completa**

Así concluye nuestra introducción en los medios de comunicación-res. Se espera que usted no va a entender mucho de lo que acaba de ocurrido- es sólo la intención de caer en el mundo tan pronto como sea posible. Algunos de los conceptos introducidos aquí se le dará mucho más tiempo en pantalla después. Algunos de ellos eran simplemente necesario para la configuración y nunca serán llevados de nuevo.

Ahora que acaba de entrenamiento.

¿Teniendo problemas? Envíame una pregunta por correo electrónico a jeffrey@emberscreencasts.com. Te voy a enviar una respuesta, y crear un FAQ con respuestas a las preguntas más comunes

También puede comprobar fuera de los repositorios de GitHub para [code¹ de cada capítulo](https://github.com/jeffreybiles/chapter-by-chapter-game) o [el acabado](https://github.com/jeffreybiles/pac-man) [addon²](https://github.com/jeffreybiles/pac-man).

¹<https://github.com/jeffreybiles/chapter-by-chapter-game>

²<https://github.com/jeffreybiles/pac-man>

**Formación**

El desierto se extiende más lejos que se puede ver. Sólo el remolino de polvo de vez en cuando interrumpe la monotonía.

“Así que este es el mundo exterior.”

Elder asiente Matteo.

“¿Es todo así?”

"La mayor parte. Aún parches no hemos llegado a todavía.”‘Así que era diferente antes ...’

“Cuando era un niño, tuvimos exuberantes selvas, lagos tranquilos ... la comida ... no necesitamos a pellets de minas en ese entonces.”

“¿Cuánto tiempo ha pasado desde que has visto esto ... la comida?”

Una luz se enciende en los ojos del anciano. “Vi una cereza año pasado. Durante la visita a la burbuja vecina.

Fue hermoso…. pero lo suficiente de reminiscencias de un anciano. Si esta burbuja va a ser alimentado, necesitamos más

operadores de APA. ¿Estás listo?"

Que a su vez su imaginación lejos de lo que podría ser una cereza, y mira a su PAC. Su Pellet Adquisición de la cápsula. Usted ha estado procesando los pellets durante años, pero al ver un PAC de cerca por primera vez, sigue siendo una experiencia impresionante. “Preparado como nunca seré.”

Que se obtiene en la esfera amarilla y dejar que los cables se envuelven. Matteo cierra la escotilla. “Vamos a que se familiarice con este artilugio, ¿de acuerdo?”

7

**2: ponerse al día,**

El último capítulo se arrojó de cabeza en un gran número de nuevas tecnologías. Me alegro de que lo hizo salir bien!

Esta es la parte del libro en el que se introduce un montón de personajes, pero no se sabe mucho acerca de ellos todavía. No se preocupe, es muy probable que! Recordar lo que pueda, y volver a este capítulo, si es necesario.

En este capítulo vamos a concentrarse en el componente de pac-man, repasando diferentes partes del código hasta que sea (en su mayoría) ya no misterioso.

**componentes**

En app / compents / PAC-man.js, Verá el siguiente código:

* Ember.Component.extend ({
* ...

3 })

Esto significa que es un PacMan Componente- Ember hereda todas las propiedades de un componente Ember, al tiempo que añade su propio. Un componente Ember es básicamente un contenedor que está sellado desde el mundo exterior, a excepción de si se asoman algunos agujeros en ella. La mayoría de los programas del mundo real tendrán que meter un par de agujeros, pero no estamos metiendo todos los agujeros en este momento, por lo que todos tenemos que tener en cuenta es en

app / componentes / PAC-man.js y app / plantillas / componentes / PAC-man.hbs (y estilo / app.css).

Como alguien que escribe un libro de introducción, esto es muy conveniente. Usted puede haber oído acerca del router Ember y lo revolucionario e importante labor que ... bueno, que va a escribir un juego completo de Ember y nunca tocarlo. Lo mismo con Inicializadores, controladores, servicios, e innumerables otras características Ember que son necesarias en muchas aplicaciones, pero también son confusos para los recién llegados. Tenemos la oportunidad de tomar ventaja de las partes que son útiles y dejar las cosas más confusas para el segundo libro.

Si desea obtener más información acerca de los componentes, [aquí hay una serie de videos sobre them³](https://www.emberscreencasts.com/tags/components).

Por supuesto, los componentes no siempre consisten en sólo un javascript Archivo- que puede tener un manillar asociados (que también puede consistir en sólo un archivo de manillar, aunque eso no sucede en este tutorial).

³<https://www.emberscreencasts.com/tags/components>

8

|  |  |
| --- | --- |
| 2: ponerse al día, | 9 |

**Bigote daliniano**

Un archivo manillar es algo así como un archivo html, pero se puede hacer un poco de programación limitada en ella. Puede variables de entrada, su uso si las declaraciones, y más (pero no mucho más). Cada plantilla está asociada con una clase de Ember, y que es donde se pone las variables que utiliza.

Más adelante vamos a utilizar más de estas características, pero por ahora todo lo que vamos a hacer es crear el canvas de HTML5 (usando simplemente vieja html).

* *<! - en app / templates / componentes / PAC-man.hbs ->*
* <**lona** carné de identidad="myCanvas" anchura="800" altura="600"> </**lona**>

**Lona**

Lona está bien named- es como el lienzo de un pintor. Se puede pintar sobre ella ... entonces se puede pintar sobre ella de nuevo, la pintura parcialmente sobre lo que había dibujado anteriormente. Y entonces, al igual que el lienzo de un verdadero pintor⁴, Puede crear un grupo de robots que salpicaduras de pintura sobre el lienzo en intervalos precisos con el fin de crear la ilusión de movimiento.

Este lienzo por defecto es blanco puro, pero hemos decidido que broncearse un poco uso de CSS.

**css**

app / estilos / app.css es donde vamos a almacenar la información de estilo. Mientras CSS es un potente herramienta, en este libro estamos básicamente sólo va a usar para cambiar las cosas colores bonitos, así que no hay conocimiento previo necesario (y, probablemente, sin conocimiento que se obtendrá). Usted puede simplemente copiar y pegar cualquier css se presentan directamente enapp / estilos / app.css.

Esto es lo que copiar-pegar en el capítulo anterior:

* **#myCanvas** {
* color de fondo: # EDC9AF;

3 }

Convierte el lienzo de un bonito color tostado.

CSS significa "Cascading Style Sheets, y es mucho más importante en una aplicación web normal que en pac-man. Si usted está interesado en aprender más,[haga clic aquí para un gran](https://www.codeschool.com/paths/html-css) [courses⁵ introductoria](https://www.codeschool.com/paths/html-css) (suscripción necesario).

⁴Si usted es inmensamente rico, un brillante maquinista, y no se preocupan por las leyes de la física.

⁵<https://www.codeschool.com/paths/html-css>

|  |  |
| --- | --- |
| 2: ponerse al día, | 10 |

**contexto de la lona y dibujo**

Por supuesto, no sólo queremos que el color bronceado ... queremos un círculo!

Ahí es donde el siguiente código entra en juego

* **dejar** lona=documento.getElementById ("myCanvas");

2 **dejar** CTX = canvas.getContext ("2D");

3

4 ...

5

6 ctx.fillStyle = '# 000';

7 ctx.beginPath ();

8 ctx.arc (x, y, radio, 0, Mates.PI \* 2, **falso**); 9 ctx.closePath ();

1. ctx.fill ();

La primera línea agarra el lienzo, a continuación, la segunda línea se hace el contexto 2d, que llamaremos CTX para abreviar.

Los últimos cinco líneas están dibujando el círculo en el contexto 2d del lienzo. No es necesario comprenderlos. Pero sí es necesario saber quédejar es.

**Dejar**

dejar está muy bien nombrado. Tome este ejemplo:

* **dejar** chapterNumber= 2;

Leer en voz alta: 'Que chapterNumber igual a dos'. Eso es más o menos lo que hace.chapterNumberahora es una variable igual a dos. Otra forma de pensar de esto es “la asignación de valores a las variables”, donde2 es el valor y chapterNumber es la variable.

Si se ha programado en javascript antes, usted sabrá acerca var. dejares similar, pero tiene propiedades ligeramente diferentes de alcance, que son generalmente menos confusa. Si no se ha programado en javascript antes, puede pasar por alto que la última frase.

Así que justo antes de sacar nuestro círculo que tenemos un montón de declaraciones vamos, y luego usamos esas variables en el dibujo.

|  |  |
| --- | --- |
| 2: ponerse al día, | 11 |

* **dejar** X= 50;

2 **dejar** y = 100;

3 **dejar** radio = 20;

4 ctx.arc (x, y, radio, 0, Mates.PI \* 2, **falso**);

es equivalente a

* ctx.arc (50, 100, 20, 0, Mates.PI \* 2, **falso**);

Nombramos las variables porque es más fácil de leer. También es posible que desee extraer más adelante (por ejemplo, el siguiente capítulo).

dejar es una característica de la ES2015, que no es compatible con todos los navegadores ... ¿por qué estamos usando? Debido a que estamos usando Babel, una ES2015 + transpiler. Eso significa Babel hace su código que se escribe en el ES2015 (o ES2016, ES2017, etc.) y lo convierte en código que (casi) todos los navegadores pueden leer (lo siento, los usuarios de IE7).

“Pero espera,” usted pregunta, “¿Cómo lo instalo Babel?”

Ya está instalado. Que empezó a usar en el momento en que ha generado el proyecto. Esto es (parte de) la magia de la brasa-cli.

**didInsertElement**

didInsertElement Es su forma de contar el componente Ember “bueno, después de que haya terminado poniendo esto componente en la pantalla, quiero que hagas esto “.

“Poner este componente en la pantalla” generalmente significa “mostrar lo que está en el archivo de manillar”. En este caso, el HTML5 Canvas.

Aquí está nuestro código completo:

* didInsertElement: **función**() {
* **esta**.drawCircle ();3 },

Así que lo que está diciendo es “cada vez que haya terminado de poner este componente en la pantalla, dibujar un círculo”.

Usted puede preguntarse qué pasa con toda la función() y las llaves {} significa ... bueno, esas son partes de la definición y el uso de funciones.

**funciones**

Las funciones son una gran manera de reutilizar el código. Definimos una vez, y luego no tener que escribir ese código de nuevo.

Aquí nos está dibujando un círculo:

|  |  |
| --- | --- |
| 2: ponerse al día, | 12 |

* **esta**.drawCircle ();

Aquí nos está dibujando dos círculos:

* **esta**.drawCircle ();

2 **esta**.drawCircle ();

Es fácil, porque definimos el drawCirclefunción. Vamos a entrar en las marcas divertidos tarde.

Aquí está el formato básico de la definición de una función:

* nombre de la función: **función**() {
* ...

3 }

Hay otro ligeramente mejor manera de definir las funciones que hay de nuevo en ES2015. Vamos a esperar para introducir esta técnica, debido a la introducción de ahora haría que la introducción de los hashes en el siguiente capítulo más confuso.

Aquí nosotros es la definición de la función drawCircle:

* drawCircle: **función**() {
* **dejar** lona=documento.getElementById ("myCanvas");

3**dejar** CTX=canvas.getContext ("2D");

4**dejar** X= 50;

5**dejar** y= 100;

6**dejar** radio= 20;

7

8ctx.fillStyle = '# 000';

9ctx.beginPath ();

1. ctx.arc (x, y, radio, 0, Mates.PI \* 2, **falso**);
2. ctx.closePath ();
3. ctx.fill ();
4. },

Todo en el medio { y } se ejecute cuando llamamos drawCircle.

Estamos definiendo la función drawCircle en el componente. Una forma de pensar en ello quiere decir que el componente ahora sabe cómo dibujar un círculo. Otra forma más elegante que lo pienso es 'el drawCircle función asignada al ámbito componente'.

|  |  |
| --- | --- |
| 2: ponerse al día, | 13 |

Ámbito de aplicación puede ser una palabra miedo, pero aquí está una manera básica de pensar en ello. Si usted está en su sala de estar, y se dice “me gustaría sentarse en el sofá”, que no tiene que especificar cuál de los millones de sofás que está sentado en. Estás en el ámbito sala de estar, así que cuando usted está tratando de pensar en los sofás, el que está en la sala de estar viene a la mente en primer lugar. Es la misma razón por la que si usted habla con alguien en los Estados Unidos sobre “la guerra civil”, que van a pensar de inmediato de la guerra civil americana, no la Guerra Civil española, la revolución americana, o la Guerra Civil de Marvel Comic (tm) . Eso es porque a cuyo ámbito de los Estados Unidos⁶.

Volvamos a los círculos en los que se acercaban y hablar de esas marcas divertidas:

* **esta**.drawCircle ();

esta, En este caso, significa 'el alcance componente'. Hey, ahí es donde almacenamos la función drawCircle. ¡Que afortunado!

this.drawCircle, sin el (), Devolvería el código de la función. Cuando añadimos el () eso 'llamadas' la función, lo que significa que se ejecuta el código de la función. Y ese código dibuja el círculo.

didInsertElement es un tipo especial de función llamada un 'gancho'. ¡No se preocupe por ahora, sólo cuenta de que lo definimos usando los mismos patrones que utilizamos para definir

drawCircle

**importación y exportación**

Una última cosa es la funcionalidad de importación y exportación. Este es el sistema de módulos ES2015 en el trabajo! Y quéimportardice que es “Voy a necesitar algo de código. Dame el código “.

* **importar** Ember desde'ascua';

Esto dice “Ir al lugar (biblioteca) conocido como 'brasa', tomar la cosa por defecto allí, y lo llaman 'Ember'”.

Entonces tenemos la siguiente línea:

* **por defecto de exportación** Ember.Component.extend ({
* ...

3 })

Esto dice “Cada vez que alguien pide algo de este archivo (y no especifica ninguna cosa en particular), les dan el Componente estamos definiendo”.

Si quisiéramos obtener PacMan en otro archivo, tendríamos que decir:

⁶(Bueno, tal vez algunos nerds aquí están en el ámbito de Marvel Comics. No voy a juzgar.)

|  |  |
| --- | --- |
| 2: ponerse al día, | 14 |

* **importar** PacMan desde'Pac-man / componentes / pac-man';

Así que nos vamos a la biblioteca pac-man (nuestro proyecto), entra en la carpeta componentes, y obtener el componente denominado pac-man.

Hay otras maneras de utilizar las importaciones y exportaciones, pero no los necesitará en este libro.

[Haga clic aquí para más información sobre ES2015 modules⁷](https://www.emberscreencasts.com/posts/62-es2015-modules-import-export)

**El sentirse bien todavía?**

Hemos aprendido un montón de nuevos conceptos en este capítulo: Componentes, manillares, funciones, y mucho, lona, ​​CSS y Módulos ES2015. El resto del libro se basará en estos conceptos (e introduce otros nuevos, aunque a un ritmo más lento).

Ahora de vuelta a nuestra historia.

⁷<https://www.emberscreencasts.com/posts/62-es2015-modules-import-export>

**Controlar**

El primer día en el PAC se pasó a familiarizarse con las diferentes pantallas. Que podría haber hecho muy bien en un escritorio, pero se sospecha que parte de la formación se está acostumbrando a estar en el PAC.

El PAC no era tan estrecha dentro como lo habría adivinado por mirarlo, pero sin duda, diferente a los escritorios de ingeniería que estaban acostumbrados. Su torso y las extremidades, incluso la cabeza, quedaron atrapados en su propia red de cables. Se podía moverlos lentamente en cualquier dirección, dentro de unos límites, pero cualquier cosa repentina causada a tensar. Matteo acababa de decir, “Te alegrarás de este una vez que empezamos con el movimiento. Y las colisiones “.

Por ahora sólo era desorientador. Afortunadamente la pantalla se quedó en frente de su cara, dando una especie de familiaridad con el procedimiento. Usted ha pasado la mayor parte de su vida de trabajo delante de una pantalla.

Así que ayer había sido las pantallas, y hoy sería los botones.

Al acercarse al campo de entrenamiento, se ve que el pastor Matteo está hablando con otra persona. Cada vez más cerca de empezar a reconocer a sí Terrance Mayhew. Terrance, el hombre cuyos descubrimientos planteadas eficiencia PAC 14%. Una de las cinco personas con logros significativos tanto en la ingeniería y el cuerpo de PAC (a pesar de que había comenzado en el PAC y desplazado a la ingeniería, donde era ahora mucho más famosa). El más honrado de la 'Ahl al-Kitab.

La conversación cesa cuando llegue.

“Salam”, dice Matteo, “¿Estás listo para un experimento?”

Asientes, zumbido débilmente con nerviosismo. El dolor en las extremidades se olvida en la emoción.

“Terrance y yo hemos estado trabajando juntos en una mejora para el proceso de formación, uno que podría disminuir la desorientación y la muerte que a veces acompaña a esta etapa de la formación. Pensamos que, con sus antecedentes, sería el sujeto de prueba perfecta “.

Asientes nuevo. La disminución de la muerte suena bastante bien. Y si eso significa que llegue a trabajar en un proyecto con Terrance ...

También comienza a entender lo grande que un acuerdo Matteo está en el mundo PAC. Terrance podría trabajar con cualquier persona, y fue Matteo. Y usted tiene que trabajar con ambos. Tal vez la citación no fue una maldición después de todo.

“¿A dónde voy?” Le pregunte.

Matteo apunta a la PAC. Que paso en, inmóvil durante los pocos segundos que le toma a los cables para afianzarse.

“Vamos a empezar con la forma de controlar la PAC”.

15

**3: Entrada**

En este capítulo, vamos a aprender cómo tomar la entrada de teclado. No se va a mover todavía, pero vamos a estar mostrando el resultado en el registro.

**Este Código robar**

Entrada ha sido históricamente difícil en javascript ... pero por suerte, otras personas han hecho todo el trabajo duro por nosotros! Sólo necesitamos una manera de conseguir su trabajo duro en nuestro proyecto. Aún más, afortunadamente, otras personas han diseñado maneras fáciles para nosotros para tomar su código! Es realmente un mundo maravilloso.

No estamos realmente “toma” el código, ya que se da libremente. Estos son llamados con mayor precisión las herramientas de “código compartido”, pero durante la duración de este libro que no va a compartir ... acaba de tomar. Pero le animo a empezar a compartir su código tan pronto como se sienta cómodo al hacerlo!

Dado que está utilizando brasa-CLI, que ya tiene acceso a varios de estos maravillosos herramientas- y tomar otra herramienta de código que los une.

Para tomar la pieza de código que necesitaremos, Tipo Ember Ember instalar entre el teclado y atajosen la línea de comandos. Ahora usted tiene la biblioteca “[Ember-teclado-shortcuts⁸](https://github.com/Skalar/ember-keyboard-shortcuts)" ¡disponible para ti!

Esto le dará el paquete NPM 'brasa entre el teclado y atajos' y el paquete glorieta 'ratonera'. NPM y la glorieta son herramientas vitales en la vida cotidiana de programación, pero no vamos a ver de nuevo en este libro.

Asegúrese de reiniciar el servidor de Ember para obtener los atajos de teclado para detener el servidor Load en la línea de comandos (CTRL + C en OSX), a continuación, reinicie (ascuas) Después de instalar el complemento.

**Llevándolo en**

Ahora puede acceder a los 'código base brasa entre el teclado y atajos' desde dentro de su proyecto, pero hay un paso intermedio antes de poder utilizar cualquiera de su código. Es necesario importarlo.

En la parte superior de nuestro componente pac-man.js, en virtud de la importación Ember, importaremos el mixin KeyboardShort-cortes.

⁸<https://github.com/Skalar/ember-keyboard-shortcuts>

dieciséis

|  |  |
| --- | --- |
| 3: Entrada | 17 |

* **importar** Ember desde'ascua';
* **importar** KeyboardShortcuts de'Brasa entre el teclado y atajos / mixins / componente';

En este código, estamos importando sólo una mixin- específicamente, el mixin que está destinado a ser utilizado en un componente Ember.

Si recuerda nuestra discusión de los módulos del capítulo 2, puede ser capaz de adivinar qué archivo en el complemento estamos importando esto desde. Si no se puede, no se preocupe-entender que esto no es el núcleo de este libro.

¿Qué es un mixin? Bueno que lo preguntas.

**mixins**

Recuerda cuando dijimos que PacMan sería “heredar” un montón de propiedades de Ember.Component? Que básicamente dice “Hola PacMan, estoy Ember.Component. Que va a ser como yo, pero no Worry puede cambiar las cosas”. Mixins decir “Hey PacMan, aquí hay un montón de nuevo código relacionado se puede utilizar si lo desea.”

¿Cómo se puede mezclar en un Mixin?

* **importar** KeyboardShortcuts de'Brasa entre el teclado y atajos / mixins / componente';

2 **por defecto de exportación** Ember.Component.extend (KeyboardShortcuts, {

* ...

4 })

Así que después de su llamada ampliar, Pero antes de la {, Se pone sus Mixins (separadas por comas). Usted puede mezclar en el mismo número Mixins como desee.

**cadena de herencia**

Ahora Pacman tiene una mezcla de cosas en él desde Ember.Component, KeyboardShortcuts y el PacMan objeto en sí mismo. ¿Qué ocurre si algo se define dos veces? En el caso general, una definición sobre el objeto en sí mismo tiene prioridad sobre una definición sobre el Mixin, que tiene prioridad sobre una definición de la clase padre. Aplicado a este caso específico, PacMan late KeyboardShortcuts, que late Ember.Component. Esto lo hemos visto ya condidInsertElement, Que se definió previamente en Ember.Component pero entonces se sobrescriben en PacMan.

**La definición de atajos de teclado**

Después de haber mezclado en el mixin KeyboardShortcuts, se puede poner el siguiente código en su

PacMan clase:

|  |  |
| --- | --- |
| 3: Entrada | 18 |

* atajos de teclado: {
* arriba: **función**() {Console.log ('arriba');}, 3 },

El efecto de esto es que cuando se pulse la tecla 'arriba', su navegador dice 'arriba' (en la consola, que te voy a mostrar cómo acceder en breve).

Vamos a repasar cada elemento de esta, a partir de los hashes.

**hashes**

Aquí está un ejemplo de un hash:

* {
* nombre de pila: "Jeffrey",

3apellido: "Bilis",

4negocio secundario: "EmberScreencasts.com",

5firstTechnicalBook: "CONSTRUIR Pacman"

6 }

La forma de leer esto es: “primero nombre es Jeffrey, apellido es Biles, negocio secundario es emberscreen-casts.com, y el primer libro técnico es CONSTRUIR Pacman”.

En términos técnicos de fantasía, se puede hablar de 'llaves' y 'valores', así como “La clave primerNombre tiene un valor de Jeffrey”, o “ 'nombre:‘Jeffrey’' es un par clave-valor”. I abrir esos términos parcialmente a ser de lujo, pero también para que pueda dar mensajes importantes como “un hash es un conjunto de pares de valores clave” y “pares de valores clave en un hash se separan por comas”.

* {
* llave: valor,
* anotherKey: 'AnotherValue'

4 }

Principiantes: Lea el último párrafo de la muestra y el código de un par de veces si es necesario.

personas de avanzada: el email cualquier explicaciones más sencillas que has encontrado.

El valor de un par clave-valor podría ser una función. Eso es lo que vamos a utilizar en nuestro hash de keyboardShortcuts.

|  |  |
| --- | --- |
| 3: Entrada | 19 |

* atajos de teclado: {
* arriba: **función**() {Console.log ('arriba');},
* abajo: **función**() {Console.log ('abajo');},

4izquierda: **función**() {Console.log ('izquierda');},

5derecho: **función**() {Console.log ('derecho');},

6 },

Ahora que ya sabe lo que un hash parece, puede que observe uno de otro de hash que hemos estado utilizando:

* **por defecto de exportación** Ember.Component.extend (KeyboardShortcuts, {

2didInsertElement: **función**() {...},

* drawCircle: **función**() {...},

4atajos de teclado: {...},

5 })

Así podemos ver que el valor de cada par clave-valor puede ser una función de hash u otra. También puede ser una propiedad más simple, como un número o una cadena.

Ahora que entendemos los hashes, vamos a ver los resultados de nuestro trabajo.

**Explotación florestal**

Se dará cuenta de que después de cada una de las direcciones en keyboardShortcuts hemos utilizado una console.log. Este dice, “Hey navegador, poner esto en la consola para que el programador puede verlo.”

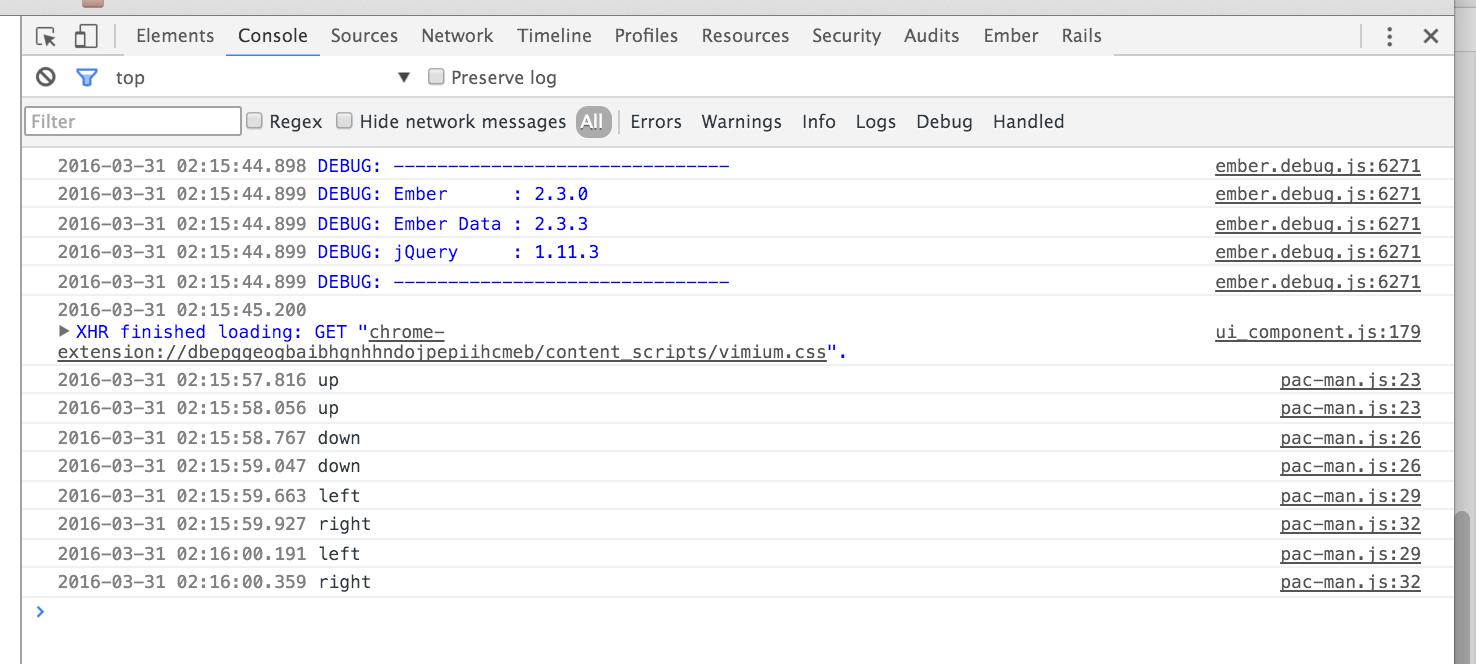
Pero cuando en el navegador no lo puso?

1. Abre Chrome.
2. Haga clic derecho en cualquier lugar de la pantalla.
3. Elija “Inspeccionar elemento” en el menú desplegable
4. O bien haga clic en la pestaña 'consola' o pulse la tecla de escape.

Ahora usted está buscando en la consola. ¡Hurra! Haga clic de nuevo en el juego, luego pulsa algunas teclas de dirección.

Si ha configurado su código como el que hemos descrito, podrás ver las instrucciones impresas.

|  |  |
| --- | --- |
| 3: Entrada | 20 |



Se puede poner casi cualquier cosa en console.log. Usted puede poner las clases, funciones, números, cadenas (lo que hay actualmente disponibles allí), hashes, y mucho más. También se puede poner más de una hora de allí, simplemente separarlos por comas.

**Resumen**

En este capítulo, nos hemos dado nuestro código algunos ... dirección. Hemos aprendido acerca de cómo importar bibliotecas externas, cómo utilizar mixins, cómo utilizar los hashes, y cómo conectarse cosas a la consola.

El siguiente capítulo utilizar nuestro botón de presionar para cambiar realmente las cosas en el mundo del juego.

**Desplazamiento**

“¿Estás listo para probar esto vivir?”, Dice Matteo. La radio crepita levemente- caparazón de la PAC interfiere mucho más de lo que cabría esperar.

“Sí”, dice usted. “Si quisiera imprimir cosas en la pantalla me hubiera quedado en la ingeniería.”

Convenientemente no se dice es que usted no tenía mucha elección. Después de que el último documento, que había sido “cordial” “invitados” a unirse al cuerpo de PAC. No se presenta por una invitación del Consejo de ancianos, no importa lo extraño o fuera de lugar que parece.

No se puede ver nada fuera de la PAC excepto en el mapa, pero se puede oír Matteo y Terrance hablando lejanamente en la radio.

“El cien en una fila sin ningún problema. Estamos bien “, dijo Matteo.

“Sacudida leve en materia de cinco.”

“Bueno, tenemos que hacer que lo entrenó, e incluso con 'ligera sacudida', esto es mucho más seguro que el uso de los PAC de producción.”

“No los viejos PAC producción”.

Matteo gruñó. “Usted sabe por qué los que tuvo que ser abandonado.”

Hay silencio durante un rato, y luego el PAC a su alrededor comienza a temblar. Se instala rápidamente a un zumbido bajo.

La voz de Matteo viene en el radio- más fuerte que cuando eras escuchas. Mucho más fuerte. Que a su vez el volumen hacia abajo. “Muy bien, usted debe ser bueno para ir. Cuidado de no pulsar demasiados botones a la vez. Esta cosa es ... poderosa “.

Usted cautela pulsa el botón “arriba” y, después de un rápido movimiento, se ve que tu lugar en el mapa ha cambiado. “¿Acabo de ir al norte?”

“Ver”, dice Terrance, “Ni siquiera podía decir que estaba sucediendo.”

“Tomar la vuelta para dar una vuelta, ver cómo se siente”, dice Matteo. “Trate de no ir fuera de la pantalla ... los equipos no saben dónde se encuentra, y la capacitación en APA se apagará. Vamos a tener que remolcar de vuelta “.

Pulsa el botones- tal vez un poco demasiado rápido al principio, ya sólo unos pocos clics se toman casi hasta el borde del radar. Mejor ir a otro lado por un tiempo.

Es una increíble pieza de tecnología, pero tiene algunas limitaciones extrañas. Decide solicitar. “Por lo tanto, no acabo de recibir cuatro direcciones? ¿Qué pasa si quiero ir a una distancia diferente?”

“Eso no es una limitación en las cuevas. No, es casi una bendición “.

21

|  |  |
| --- | --- |
| Desplazamiento | 22 |

Usted había visto diagramas de las cuevas antes, pero por supuesto que los ángulos rectos y un espaciado uniforme eran una simplificación. Tal vez no lo eran. Pero sería extraño, un mundo en forma tan, un mundo de rejilla sin curvas o variación.

Se nota que la máquina ya no tiembla cuando se pulsa el botón, y darse cuenta de que se haya ido al perímetro.

Los ancianos han dado cuenta de que lo haga. Matteo está maldiciendo.

"¿Ver? Quitarle las votaciones áspera y consiguen descuidados “.

“Te lo prometo”, dice Terrance, “Los beneficios serán mayores que los inconvenientes.” “Ellos mejor.”

“¿Cuántos días hasta que esté listo?”

“Para llegar a 238? Una semana, si la otra tecnología que hemos llegado con obras como se esperaba “.

“Y ¿cuánto tiempo hasta que lleguen aquí?”

“Usted sabe que es clasificada.”

“Usted sabe que no importa si no tenemos éxito.” Un largo silencio.

“Están barriendo a través de la India en este momento. Todo depende de cuánto tiempo los niños pueden mantenerlos ocupados “.

Un silencio más tiempo.

La voz de Matteo entra en juego otra vez, en voz alta. Esta vez él significa para ser escuchado. Usted lucha para bajar el volumen.

"Casi estámos allí. Vamos a tirar de vuelta. Ah, y vamos a ser la aceleración de su formación. Vamos subterránea “.

**4: Movimiento**

En este capítulo vamos a hacer nuestro movimiento PAC. Para nuestro héroe fue simplemente presionando un par de botones, pero a medida que los programadores sabemos que hacer las cosas simples que puede ser un trabajo duro.

**propiedades**

Recuerde nuestra función drawCircle?

* drawCircle: **función**() {
* **dejar** lona=documento.getElementById ("myCanvas");

3**dejar** CTX=canvas.getContext ("2D");

4**dejar** X= 50;

5**dejar** y= 100;

6**dejar** radio= 20;

7

8ctx.fillStyle = '# 000';

9ctx.beginPath ();

1. ctx.arc (x, y, radio, 0, Mates.PI \* 2, **falso**);
2. ctx.closePath ();
3. ctx.fill ();
4. }

Es una muy buena función, si me permite decirlo. Es realmente grande en el dibujo un círculo negro en [50, 100] con un radio de 20. Sin embargo, queremos hacer algo más que dibujar un círculo. Vamos a tomar algunas medidas para hacernos una especie de multi-círculo.

En primer lugar, vamos a tener que separar donde almacenamos los valores de X, yy radio. En lugar de definirlos dentro de ladrawCircle función, vamos a ser almacenarlos en nuestra PacMancomponente. De esa manera podemos acceder (y cambiar) desde otro lugar. (Nosotros también vamos a almacenarsquareSize en lugar de radio, Para nuestra conveniencia en un futuro cercano)

23

|  |  |
| --- | --- |
| 4: Movimiento | 24 |

* *// ...*
* **por defecto de exportación** Ember.Component.extend (KeyboardShortcuts, {
* X: 50,
* y: 100,
* squareSize: 40,
* drawCircle: **función**() {
* **dejar** lona=documento.getElementById ("myCanvas");

8**dejar** CTX=canvas.getContext ("2D");

9**dejar** X= **esta**.obtener('X');

1. **dejar** y= **esta**.obtener('Y');
2. **dejar** radio= **esta**.obtener('SquareSize')/ 2;
3. ctx.fillStyle = '# 000';
4. ctx.beginPath ();
5. ctx.arc (x, y, radio, 0, Mates.PI \* 2, **falso**);
6. ctx.closePath ();
7. ctx.fill ();
8. }
9. *// ...*
10. }

Así que aquí estamos almacenando nuestras coordenadas y squareSize en el hash (Pacman Componente), de la misma manera que estamos almacenando la función drawCircle en el hash. Dentro de la función drawCircle, estamos acceder a los x, y, y las propiedades squareSize utilizando la función de 'conseguir'.

Asi que this.get ( 'x')significa “En este componente hay algo llamado x, y, por favor obtener su valor”. Por lo que el programa va diligentemente en esta instancia de componente y encuentra el valor asociado con la tecla 'x'.

**actualización de las propiedades**

Que acabamos de aprender cómo conseguir propiedades en el hash, y esto es progreso. Sin embargo, para mover la PAC que vamos a tener que aprender a cambiar esas propiedades.

La forma más general de hacer esto es con el conjunto método.

* **esta**.conjunto('X',150)

Ese pedazo de código se establecerá el valor de X en esta (El componente) a 150.

Aquí hay una línea de código que no sólo se configurará el valor de x, pero cambiarlo cada vez que la línea se llama:

|  |  |
| --- | --- |
| 4: Movimiento | 25 |

* **esta**.conjunto('X', **esta**.obtener('X')+ **esta**.obtener('SquareSize'))

Esta toma el valor de x y después lo cambia por el tamaño de un cuadrado (el diámetro de nuestra PAC).

Vamos a ser la suma y resta de los valores un poco, por lo que vamos a utilizar algunos de Ember incrementProperty y decrementProperty.

A veces se denominan “métodos de conveniencia”. métodos de conveniencia son funciones que no hacen nada realmente nuevo y emocionante (reemplazando a menudo sólo una o dos líneas de código con una sola línea de código más corto), pero son suficientes expresiones comunes que tiene sentido para proporcionar una forma rápida para llevarlas a cabo.

Por lo que la línea anterior sería

* **esta**.incrementProperty ('X', **esta**.obtener('SquareSize'))

A pesar de que es casi tan larga como lo era antes, es mucho más fácil de entender de un vistazo.

**Traslado de la PAC**

Así que vamos a utilizar este nuevo conocimiento para mover nuestra PAC. Vamos a actualizar el código dederecho en nuestro hash de keyboardShortcuts:

* atajos de teclado: {
* arriba: **función**() {Console.log ('arriba');},
* abajo: **función**() {Console.log ('abajo');},

4izquierda: **función**() {Console.log ('izquierda');},

5 derecho: **función**() { **esta**.incrementProperty ('X', **esta**.obtener('SquareSize')); **esta**.\ 6 drawCircle ();},

7 },

Así que cuando pulsamos el derechotecla de flecha el valor de x se incrementará por el tamaño de un cuadrado (el diámetro de la PAC). También vamos a volver a dibujar el círculo llamandodrawCircle. sin llamardrawCircleque no sería capaz de ver los cambios. Esto tiene el efecto visual de mover el círculo de una “cuadrado” a la derecha de donde estaba antes.

Una nota rápida sintaxis: el punto y coma suele ser opcional, pero se requiere cuando se separan dos instrucciones diferentes que están en la misma línea.

Vamos a hacer que para todas las otras direcciones.

|  |  |
| --- | --- |
| 4: Movimiento | 26 |

* atajos de teclado: {
* arriba: **función**() { **esta**.incrementProperty ('Y', -1 \* **esta**.obtener('SquareSize')); Thi\ 3 s.drawCircle ()},

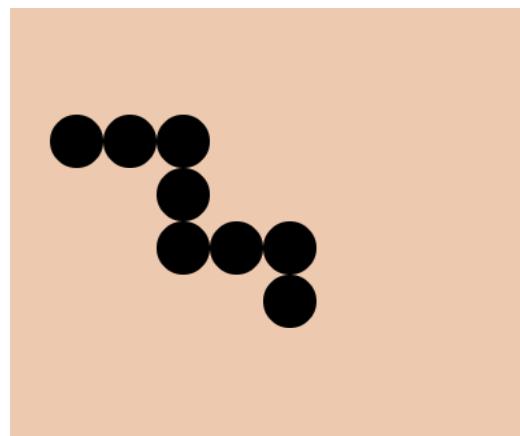
4 abajo: **función**() { **esta**.incrementProperty ('Y', **esta**.obtener('SquareSize')); **esta**.\ 5 drawCircle ()},

6 izquierda: **función**() { **esta**.incrementProperty ('X', -1 \* **esta**.obtener('SquareSize')); t\ 7 his.drawCircle ()},

8 derecho: **función**() { **esta**.incrementProperty ('X', **esta**.obtener('SquareSize')); **esta**.\ 9 drawCircle ()},

1. },

Trate de hacer esto y usted será capaz de hacer el movimiento PAC por la pantalla. Sin embargo, se verá que mientras que el nuevo círculo se ve arrastrado, los antiguos círculos se quedan. Vamos a cuidar de ello.



**Borrado de la pantalla**

Vamos a crear una pantalla clara función:

|  |  |
| --- | --- |
| 4: Movimiento | 27 |

* pantalla clara: **función**() {
* **dejar** lona=documento.getElementById ("myCanvas");

3**dejar** CTX=canvas.getContext ("2D");

4**dejar** ScreenWidth= 800;

5**dejar** ScreenHeight= 600;

6

7ctx.clearRect (0, 0, ScreenWidth, ScreenHeight)

8 }

Al igual que en nuestro drawCirclefunción, vamos a agarrar el lienzo y luego coger el contexto 2d. A continuación, vamos a tomar una acción en el contexto- en este caso, la limpieza de un rectángulo del tamaño de la pantalla.

Ahora sólo tenemos que llamar pantalla clara cada vez que movemos el PAC:

* atajos de teclado: {
* arriba: **función**() { **esta**.incrementProperty ('Y', -1 \* **esta**.obtener('SquareSize')); Thi\ 3 s.clearScreen (); **esta**.drawCircle ()},

4 abajo: **función**() { **esta**.incrementProperty ('Y', **esta**.obtener('SquareSize')); **esta**.\ 5 pantalla clara(); **esta**.drawCircle ()},

6 izquierda: **función**() { **esta**.incrementProperty ('X', -1 \* **esta**.obtener('SquareSize')); t\ 7 his.clearScreen (); **esta**.drawCircle ()},

8 derecho: **función**() { **esta**.incrementProperty ('X', **esta**.obtener('SquareSize')); **esta**.\ 9 pantalla clara(); **esta**.drawCircle ()},

1. },

Cuando se ejecuta el programa que será capaz de mover el PAC y el pantalla clara función borra sus antiguas posiciones.

Por lo que funciona, pero el código no sólo es desordenada y repetitiva, sino que también se repite.

Vamos a arreglar eso antes de seguir adelante.

**La extracción de los métodos**

Vamos a tomar nuestro código desordenado, repetitivo y extraer un método de ella. Haremos esto al notar lo que los patrones son y abstraer a la basura. Llamaremos a nuestro nuevo métodomovePacMan.

|  |  |
| --- | --- |
| 4: Movimiento | 28 |

* movePacMan: **función**(Dirección, cantidad) {
* **esta**.incrementProperty (dirección, cantidad);
* **esta**.pantalla clara();
* **esta**.drawCircle ();5 },

movePacMan toma los tres métodos que llamamos cada vez que nos estamos moviendo nuestra PAC y las pone en un método. entonces podemos utilizar ese método cuando estamos definiendo qué hacer después de las pulsaciones de teclado.

* atajos de teclado: {
* arriba: **función**() { **esta**.movePacMan ('Y', -1 \* **esta**.obtener('SquareSize'));},

3abajo: **función**() { **esta**.movePacMan ('Y', **esta**.obtener('SquareSize'));},

4izquierda: **función**() { **esta**.movePacMan ('X', -1 \* **esta**.obtener('SquareSize'));},

5 derecho: **función**() { **esta**.movePacMan ('X', **esta**.obtener('SquareSize'));}, 6 },

No sólo es que más corto, pero también es mucho más fácil de entender. Como un beneficio adicional, si queremos cambiar el proceso de mover el PAC entonces sólo tenemos que cambiarlo en un solo lugar (lamovePacMan Método) en lugar de cuatro lugares.

Vamos a seguir adelante y hacer una extracción más.

**Una propiedad computarizada**

Hemos conseguido contexto el lienzo dos veces, y probablemente vamos a estar recibiendo aún más en el futuro, por lo que vamos a hacer que sea más fácil para nosotros hacer eso.

* CTX: Ember.computed (**función**() {
* **dejar** lona=documento.getElementById ("myCanvas");

3**dejar** CTX=canvas.getContext ("2D");

4**regreso** CTX;

5 }),

Hay dos cosas nuevas aquí, y que trabajan en concierto.

El primero es el regresodeclaración, lo que hace esta función diferente a las que hemos creado antes. Las funciones anteriores han hecho algo (hecho un cambio en el mundo), pero no cambia nada, y en lugar de simplemente devuelve un valor.

El segundo es el Ember.computedmétodo que envuelve nuestra función. Esto lo convierte en 'propiedad calculada'. Vamos a aprender más sobre esto más adelante, pero lo que significa para nosotros es que ahora podemos tratarCTX como una propiedad en lugar de como un FunciÃ³n- como X en lugar de como drawCircle.

|  |  |
| --- | --- |
| 4: Movimiento | 29 |

Así, mientras que una función regular se llama así: this.drawCircle ().

Una propiedad computarizada se recupera de esta manera: this.get ( 'ctx').

Vamos a utilizar esta propiedad calculada en ClearScreen y drawCircle.

* pantalla clara: **función**() {
* **dejar** CTX= **esta**.obtener('CTX');

3**dejar** ScreenWidth= 800;

4**dejar** ScreenHeight= 600;

5

6ctx.clearRect (0, 0, ScreenWidth, ScreenHeight)

7 }

* drawCircle: **función**() {
* **dejar** CTX= **esta**.obtener('CTX');

3**dejar** X= **esta**.obtener('X');

4**dejar** y= **esta**.obtener('Y');

5 **dejar** radio = **esta**.obtener('SquareSize')/ 2; 6

7ctx.fillStyle = '# 000';

* ctx.beginPath ();
* ctx.arc (x, y, radio, 0, Mates.PI \* 2, **falso**);

10ctx.closePath ();

11ctx.fill ();

12 }

**Resumen**

En este capítulo hemos aprendido cómo mover nuestra PAC.

Para ello tuvimos que aprender acerca de las propiedades, que podemos recuperar con this.get y cambiar con este conjunto (O el método de conveniencia this.incrementProperty).

También hemos creado nuestros propios métodos, tales como pantalla clara y movePacMan, Que utilizamos para separar el código en bloques lógicos, reutilizables, fácilmente con nombres.

Finalmente, conseguimos nuestra primera muestra de computada propiedades- un concepto muy útil que estaremos ampliando en los próximos capítulos.

En el siguiente capítulo, vamos a utilizar las redes y matrices para comenzar a preparar nuestra PAC para el mundo subterráneo de paredes-off que está destinado a habitar.

**barreras**

Su primera noche en el cuartel fue sin incidentes, pero se podía sentir los susurros.

En el comedor había sido mesas llenas de veteranos canosos. Que estabas cerca de ochenta libras de la media- la mayor parte de la masa muscular extra era, pero había un poco de acolchado adicional y una capa de tejido cicatrizal tirado en una buena medida. Estaba claro que no encajaba.

Inducciones a los PAC-hombres pasaron tres veces al año, y fueron precedidos por meses de acondicionamiento físico. Los bonos construyeron hubo el núcleo de la experiencia de PAC-MAN. Pero You- que fueron traídos en un off-meses, claramente con poca preparación.

Oraciones habían sido lonely- haciendo las observancias Dhuhr y ASR sin su equipo a tu lado era lo máximo que se había perdido ellos todavía-, pero el comedor era peor.

Sentado solo, tratando de no escuchar la conversación de los demás por encima del estruendo. Tratando de no completar los fragmentos de conversación que se oye, tratando de no hacerlas por ti. Tratando de justificar la risa como sólo una buena broma que alguien dijo.

Finalmente alguien venga y establezca su bandeja delante de usted. “Salam?”

Nos fijamos en él, entrecierra los ojos. Algo sobre esa cara. Mentalmente se quita una capa de músculo, quitas dos años de fuerte desgaste.

“Jerome?”

El sonrie. “Y dijeron mis viejos amigos no me reconocerían. Entonces, ¿qué diablos haces aquí? Pensé que eras el niño de oro de procesamiento de pellets “.

“Todavía estoy. No sabemos exactamente por qué me quieren, pero los ancianos realmente no piden mi opinión sobre el tema “.

“Bueno, siempre y cuando no matan al niño de oro que pone la investigación de oro.” “¿Ha estado continuando la tuya?”

“Me moví en las pruebas de choque. Para hacer que los PAC más fácil de manejar, menos ... brutal. Veo por su falta de ojos negros que mi trabajo ha dado sus frutos “.

"¿Brutal? Hasta ahora, la mía ha sido suave como la seda “.

“Estás en un PAC entrenamiento. Son gentle- incluso más suave que el viejo estilo, gracias a mí-, pero se comen energía. Solían ser todos así, pero ya que fuimos más y más en la tierra se hizo más difícil y más difícil de hacer que funcione, económicamente. Necesitábamos algo más eficiente ... y eso significaba sacrificar cierta calidad de vida de los conductores. De ahí que todos estos chicos que ven a su alrededor. Maltratadas héroes musculosos sangrientas que nos recibieron todas las pastillas que necesitamos “.

“¿Cuánto tiempo hasta que yo veo como eso?”

30

|  |  |
| --- | --- |
| barreras | 31 |

Jerome se encoge de hombros. “Mi más reciente prototipo tiene un cuarto de la tasa de lesiones como el estándar y cinco por ciento de la tasa de lesiones graves. Así que ... ya que antes. Pero, honestamente, no están listos para la construcción de un científico “.

El resto de la cena pasó sin incidentes, un simple intercambio entre amigos-en todo caso aquí podría ser llamado simple-pero se tomó esa línea como un reto. Jerome había sido casi tan delgada como cuando salió de la ingeniería de los PAC-hombres. Tal vez usted no estaba listo para la experiencia del hueso traqueteo lleno de vida del PAC, pero que iba a hacer su imposible por lograr lo que la misión de los ancianos tenían para usted.

Por la mañana Matteo es todo el negocio.

“Hoy vas a empezar el entrenamiento subterráneo. Se necesita algún tiempo para acostumbrarse. Las viviendas que ha labrado, redondean los bordes con el fin de satisfacer nuestra psique. Pero en las minas de pellets, es ángulos duros y fríos, noventa grados, igual separación “.

“Lo tengo, es una cuadrícula.”

“Usted dice que lo consigue, pero en realidad nadie lo consigue hasta que han estado operando allí por un tiempo.” “Entonces vamos a operar”.

Él ríe. "Me gusta tu estilo."

Entras en una habitación llena de los PAC. La mayoría están entrenando a los PAC como el que se va a utilizar hoy en día, pero algunos parecen que han sido a través del escurridor un par de veces. Matteo apunta a un PAC formación y de subir. El interior no es tan extraño como el primer avance tiempo-.

El equipo que rueda en una habitación perfectamente rectangular, y comienza el entrenamiento.

**5: Armario**

En el capítulo 4 tenemos la PAC en movimiento, pero había un problema: se puede ir fuera de la pantalla. Para Salam que significaba que su formación PAC cerrado, pero para nuestro jugador que significaba que estaban perdidos sin tener idea de dónde estaban.

En este capítulo vamos a encerrar la PAC dentro de la pantalla- pero en primer lugar, vamos a tener que hacer algunos cambios con la forma en que pensamos sobre el espacio en el juego.

**Alineación**

El primer cambio que haremos nos va a alinear más estrechamente con la lógica del juego.

Actualmente, hacemos un seguimiento de qué píxeles se coloca el PAC en- la x y Y píxeles en los que se centra el círculo. Entonces nos movemos hacia arriba, abajo, izquierda o derecha por el diámetro del círculo. Pero eso no es lo que pensamos de la misma. Pensamos en ello como una cuadrícula, y el PAC se está moviendo entre las plazas de la parrilla.

Pensando en el PAC como ocupando una plaza en una cuadrícula, y no sólo como un círculo en el espacio, toma un poco más de esfuerzo en la delantera mientras construimos la abstracción, pero nos va a ahorrar una gran cantidad de trabajo muy pronto cuando añadimos cosas como paredes y pellets y fantasmas. Empecemos.

En primer lugar vamos a cambiar la forma en que registramos la posición de partida, desde la posición de píxel pura:

* X: 50,

2 y: 100,

a la mancha en la red:

* X: 1,

2 y: 2,

No es una comparación exacta, porque el 50 y el 100 no van uniformemente en 40. Las rejillas va rejilla.

A continuación, vamos a dibujar el círculo utilizando el nuevo sistema de red:

32

|  |  |
| --- | --- |
| 5: Armario | 33 |

* drawCircle: **función**() {
* **dejar** CTX= **esta**.obtener('CTX');

3**dejar** X= **esta**.obtener('X');

4**dejar** y= **esta**.obtener('Y');

5**dejar** squareSize= **esta**.obtener('SquareSize');

6

7**dejar** pixelX=(X+1/2)\*squareSize;

8**dejar** pixelY=(Y+1/2)\*squareSize;

9

1. ctx.fillStyle = '# 000';
2. ctx.beginPath ();
3. ctx.arc (pixelX, pixelY, squareSize/ 2, 0, Mates.PI \* 2, **falso**);
4. ctx.closePath ();
5. ctx.fill ();
6. },

Nótese cómo estamos calculando pixelX y pixelY desde el X y yposiciones de la cuadrícula. Estamos añadiendo a cada medio, porquectx.arc quiere que el centro del círculo, y simplemente multiplicando por squareSize daría a la posición superior izquierda en esa cuadrícula.

No entiendo el párrafo anterior? Trate de hacer sin el medio y ver por sí mismo! La gran cosa sobre el desarrollo del juego es que se pueden realizar cambios en el código y luego ver de inmediato los efectos por sí mismo de una manera muy visual.

A continuación, vamos a echar otro vistazo a los lugares en los que estamos usando X y y. Por ejemplo, en el teclado accesos rápidos que los estamos cambiando por 40 cada vez que pulsamos una tecla.

* *// Una keyboardShortcut bajo el antiguo camino*
* arriba: **función**() { **esta**.movePacMan ('Y', -1 \* **esta**.obtener('SquareSize'));},

Con nuestro nuevo sistema, cambiándolos por 40 ya no es una gran idea. Esto es lo que el nuevo código se verá así:

* atajos de teclado: {
* arriba: **función**() { **esta**.movePacMan ('Y', -1);},

3abajo: **función**() { **esta**.movePacMan ('Y', 1);},

4izquierda: **función**() { **esta**.movePacMan ('X', -1);},

5 derecho: **función**() { **esta**.movePacMan ('X', 1);}, 6 },

La intención de esto es mucho más clara que la antigua usanza, donde tuvimos que mirar miríada pasado this.get ( 'squareSize') operaciones de búsqueda con el fin de llegar a la esencia de la misma.

En la última parte de nuestra transición a la red, vamos a hacer frente al ScreenHeight y ScreenWidth variables en pantalla clara.

|  |  |
| --- | --- |
| 5: Armario | 34 |

* ScreenWidth: 20,

2 ScreenHeight: 15,

3 pantalla clara: **función**() {

4**dejar** CTX= **esta**.obtener('CTX');

5**dejar** screenPixelWidth= **esta**.obtener('ScreenWidth')\* **esta**.obtener('SquareSize');

6 **dejar** screenPixelHeight = **esta**.obtener('ScreenHeight') \* **esta**.obtener('SquareSize'); 7

8ctx.clearRect (0, 0, ScreenPixelWidth, screenPixelHeight)

9 },

Aquí tenemos la ScreenWidth y ScreenHeight las variables de ámbito como propiedades en el componente (por lo que se pueden utilizar en otro lugar más tarde), y entonces estamos utilizando para calcular screenPixelWidth y screenPixelHeight (800 y 600, respectivamente).

Mientras que nada ha cambiado en el modo de jugar en este momento, este cambio no garantiza que mientras ScreenWidth y ScreenHeight son enteros, el área de la pantalla se ajusta a la cuadrícula.

Hasta ahora este capítulo hemos refactorizamos nuestro código para adaptarse a un nuevo modelo mental, pero no hemos aprendido nuevos trucos todavía. Eso no va a durar mucho tiempo.

**Si los estados**

Ahora que tenemos nuestro sistema de posicionamiento en forma de rejilla configurado, nuestro próximo objetivo es asegurarse de que el PAC nunca sale de la pantalla.

Hasta el momento en que se presiona un botón, que es una acción de uno por uno. Pulse hacia la derecha, a la derecha ... no importa qué. Pero ahora vamos a establecer algunas condiciones en que la acción.

Para hacer eso, estaremos utilizando una sentencia condicional. En concreto, vamos a estar utilizando laSideclaración. Aquí está el esquema básico de cómo utilizar unaSi

* *// alimentación en un valor booleano directamente*

2 **Si**(BooleanValue) {

3console.log ('BooleanValue es cierto!)

4 }

5 *// crear un booleano través de una expresión*

6 **Si**(mi número == 4) {

7console.log ('Mi número es 4')

8 }

|  |  |
| --- | --- |
| 5: Armario | 35 |

Así que la forma es Si, A continuación, una expresión booleana entre paréntesis, a continuación, algunas instrucciones entre las llaves.

Un valor booleano es cierto o falso. Una expresión booleana es una pieza de código que con el tiempo se simplifica hastacierto o falso. Sólo puede ser un valor, (una afirmación clara de la verdadera o falsa), una variable que significa la verdadera o falsa, o puede ser algo así comomyNumber == 4, Que será verdadera si mi número es el número entero 4, y falso si se trata de otra cosa.

Vamos a describir cómo vamos a asegurarse de que la PAC se mantiene dentro de los límites. Básicamente, vamos a mover el PAC, a continuación, vamos a comprobar para ver si el PAC se encuentra fuera de los límites, y si es que vamos a revertir nuestra jugada anterior. todo esto sucede antes de que se dibuja el círculo, por lo que el usuario se ve como el PAC se quedó en el mismo lugar.

Esto es lo que se vería así.

* movePacMan (dirección, cantidad) {
* **esta**.incrementProperty (dirección, cantidad);

3

4**Si**(**esta**.collidedWithBorder ()) {

5**esta**.decrementProperty (dirección, cantidad)

* }

7

* **esta**.pantalla clara();

9**esta**.drawCircle ();

1. },

Estaremos definiendo collidedWithBorder como una función en el componente pac-hombre que devuelve un valor booleano:

* collidedWithBorder: **función**() {

2**dejar** X= **esta**.obtener('X');

3**dejar** y= **esta**.obtener('Y');

4**dejar** ScreenHeight= **esta**.obtener('ScreenHeight');

5**dejar** ScreenWidth= **esta**.obtener('ScreenWidth');

6

7**dejar** pacOutOfBounds=X<0 ||

|  |  |  |
| --- | --- | --- |
| 8 | y <0 || | |
| 9 | X | > = ScreenWidth || |
| 10 | y | > = ScreenHeight |

1. **regreso** pacOutOfBounds
2. }

|  |  |
| --- | --- |
| 5: Armario | 36 |

los ||significa 'o', por lo que la lectura en voz alta sería decir “pacOutOfBounds es verdadera si x es menor que cero o Y es menor que cero o x es mayor que o igual a la anchura de la pantalla o Y es mayor que o igual a la pantalla altura". Un buen bocado, pero muy fácil de analizar cuando se utiliza símbolos.

Entonces tomamos el resultado de que la expresión booleana y establecerlo como el valor de retorno de la función. A continuación, ese resultado es lo que determina si el código en la sentencia if o no.

**colisiones totalmente funcionales**

Ejecutar ese código y tratar de salir de bounds- que siempre está detenido en la frontera.

Esta es la primera pieza de código que sólo se ejecuta conditionally- sólo se ejecuta si ciertas cosas sobre el mundo son verdaderas. Esta es una herramienta muy útil que vamos a hacer un uso liberal de a través del resto del libro.

**Encuadernación manillar**

Nuestra nueva funcionalidad se realiza para este capítulo, pero hay una parte de nuestro código que tenemos que hacer más robusto.

Ir al archivo templates / componentes / PAC-man.hbs. Usted verá esto:

* <**lona** carné de identidad="myCanvas" anchura="800" altura="600"> </**lona**>

Eso es decir que tenemos un elemento de tela con una anchura de 800 y una altura de 600. Hasta aquí todo bien! -se coincide con el PixelWidth y pixelHeight en nuestro archivo.

Pero ¿qué ocurre si queremos cambiar el tamaño de nuestro nivel? Digamos que, en lugar de ser 20 casillas de ancho y 15 plazas de altura, es de 10 casillas de ancho y 20 de altura cuadrados.

Trate de IT que pronto ver que el lienzo (donde se muestran todos los gráficos) tiene diferentes dimensiones que el espacio de movimiento permitido. Así, mientras que no podemos ir más allá de la mitad a través del lienzo a la derecha, podríamos ir hacia abajo hasta que no podemos ver el PAC más.

La solución es tener la anchura y la altura del cambio de la lona en base a la anchura y la altura de nuestro nivel.

En primer lugar vamos a calcular las propiedades en Javascript del componente:

|  |  |
| --- | --- |
| 5: Armario | 37 |

* screenPixelWidth: Ember.computed (**función**() {
* **devolver este**.obtener('ScreenWidth')\* **esta**.obtener('SquareSize');3 }),

4 screenPixelHeight: Ember.computed (**función**() {

5 **devolver este**.obtener('ScreenHeight') \* **esta**.obtener('SquareSize'); 6 }),

7 *// Asegúrese de utilizar estos valores calculados en `clearScreen`* 8 pantalla clara: **función**() {

9**dejar** CTX= **esta**.obtener('CTX');

1. ctx.clearRect (0, 0, **esta**.obtener('ScreenPixelWidth'), **esta**.obtener('ScreenPixelHeight'\
2. ))
3. },

Nada en los cálculos ha cambiado esta vez, pero los hemos de ámbito al componente.

Para los practicantes avanzados: Usando Ember.computed también almacena en caché el resultado, por lo que puede dar lugar a un pequeño aumento de velocidad (en lugar de tener que hacer el cálculo cada vez) También para los practicantes avanzados: dejé de usar cualquier claves de cómputo con el fin de dar explicaciones más fácil. Que debería estar bien en este caso, ya que estas propiedades calculadas nunca necesitarán ser re-evaluado.

A continuación, vamos a utilizar esos valores en el archivo de manillares del componente:

* <**lona** carné de identidad="myCanvas" anchura={{}} ScreenPixelWidth altura={{}} ScreenPixelHeight></ \

2 canvas>

En el código anterior, estamos estableciendo la anchura y la altura de la tela igual al valor de la screenPixelWidth y screenPixelHeight variables. En Manillar (el lenguaje de plantillas que estamos usando), cualquier cosa entre {{ y }} deja de actuar como html y comienza a actuar como código.

Este código se encuentra en el lenguaje de plantillas manillares. El lenguaje de plantillas manillares es diferente de Javascript, y mucho más simple. Para este capítulo, todo lo que necesitamos saber es que podemos mostrar una variable.

¿De dónde la variable viene? Cada plantilla está emparejado con un objeto Ember (a veces auto-generado), y tiene acceso a todas las variables y las propiedades que están en ese objeto. Esta plantilla se combina con el componente pac-man que hemos estado trabajando, por lo que tiene acceso a las propiedades screenPixelWidth y screenPixelHeight.

Seguir adelante y cambiar el ScreenWidth y ScreenHeight- el lienzo y la zona acotada será tanto cambio en la respuesta. Muy bueno, ¿sí?

|  |  |
| --- | --- |
| 5: Armario | 38 |

**Resumen**

En este capítulo nos alineamos nuestro código a una abstracción de rejilla, límites establecidos utilizando una sentencia if y la lógica booleana, y luego se sincronizan nuestra pantalla de tela y nuestro código usando manillares.

En el siguiente capítulo vamos a ir de las paredes en forma de cuadrícula a cuadrícula, configurar, y aprender acerca de las matrices y bucles.

**Dentro del laberinto**

Usted ha pasado la última hora moviendo su capacitación en APA dentro de la sala rectangular desnudo. En un primer momento se había pegado en su mayoría a la mitad, ya que la colisión con la pared exterior llevó a la formación de PAC para darle un choque en el lado en el que chocó, pero Matteo insistido en que el círculo de perímetro.

“Tienes que aprender las maniobras. Es necesario para vivir con el riesgo. Que no le guste el choque, pero cuando se obtiene en un verdadero PAC y se encuentra con una pared de bienes, podrás maldita perder.”

Después de una hora que está recibiendo golpes cada vez menos, aunque todavía necesita mucha concentración para rodear el perímetro sin caer en una pared.

Matteo te empuja en de todos modos.

“Estas áreas de entrenamiento son los únicos lugares en la tierra con una rejilla completamente claro. Maldita sea casi perfecto, si no aburrirse, pero no herido es su cosa. Antes, cuando había cosas que vale la pena subir a la superficie tuvimos que preocuparse por la densidad de la cubierta de tierra, curvas, cosas por el estilo. Era salvaje. Ahora subterránea es sólo pasadizos y paredes. Pero las paredes, que puede ser difícil si usted no es un Pac-Man “.

¿Que significaba eso? No hay tiempo para pensar en ello, a medida que continúa hablando. “Eres paredes encuentro gonna, por lo que vamos a conseguir que algunas paredes.”

Y la pared que ofrece. Se aprieta algunos botones y un pilar se levanta de la planta, un cuadrado perfecto empujado desde el suelo al techo.

“Círculo del pilar”, dice.

Mueve el PAC hacia el pilar, a continuación, iniciar la serie de movimientos que harán que usted se mueve en el circuito correcto. Hay un par de choques crueles como tratando de averiguar la mejor manera de llevar a cabo el patrón.

“No sé si estoy listo todavía”, se dice.

Matteo harumphs. “¿Qué, quieres ser perfecto antes de seguir adelante? Tal vez hacer algunas ecuaciones?”‘Creo que si pasé más tiempo ...’

“Mira, tienes la habilidad básica suficientemente bien, y esto es clasificado, pero estás en un programa de entrenamiento apretado. No es mi elección para enviarle allí abajo, pero es mi trabajo para asegurarse de que no se muere en el camino a su misión “.

“En el camino a mi misión?”

"Sí. El camino de vuelta, no es tan importante, siempre y cuando alguien tiene los resultados, sino que vamos a llamar a un objetivo ambicioso “.

“Así que usted está pensando en mí morir ahí abajo?”

39

|  |  |
| --- | --- |
| Dentro del laberinto | 40 |

“Eso depende de usted- me gustaría que usted pueda sobrevivir, orgullo profesional como entrenador y todo eso, pero no, no es misión crítica que sobreviven en el camino. Todo depende de la fuerza con la que desea trabajar “.

¿Es esta una ... Palabras de ánimo? Los militares son extraños.

“De todas formas, vas a conseguir algunos golpes desagradables en el entrenamiento, pero después de una cierta cantidad de tiempo practicando los conceptos básicos es más eficiente que ir para las habilidades que se basan en los conceptos básicos de una manera interesante. Si se siente cómodo, es seguro para seguir adelante. Para usted, si usted siente que está entrando en una rutina, nada ni siquiera cerca de una rutina, entonces la única opción segura es aprender más. No aprender le conseguirá matado “.

"¿De Verdad?"

Se encoge de hombros. "Probablemente. Ahora ve a la formación “.

Se traza un circuito alrededor de la columna. Entonces lo haces de nuevo. Y otra vez.

**6: Las barreras internas**

En el capítulo anterior hemos creado nuestra PAC en una estructura en forma de rejilla y se bloquean para que no pueda salir del área de juego. Pero pac-man se trata de mucho más que simplemente dando vueltas en una habitación acolchada, también hay algunas barreras internas que se pueden ejecutar en. En este capítulo se trata de levantar esas barreras.

**Dibujo una pared**

Vamos a echar un vistazo a lo que se necesitaría para dibujar una pared:

* **dejar** squareSize= **esta**.obtener('SquareSize');

2 **dejar** CTX = **esta**.obtener('CTX');

3 ctx.fillStyle = '# 000';

4

5 **dejar** pared = {X: 1, y: 1}

6 ctx.fillRect (wall.x \* squareSize,

|  |  |
| --- | --- |
| 7 | criajo \* squareSize, |
| 8 | squareSize, |
| 9 | squareSize) |

En primer lugar cogemos nuestra squareSize y contexto y asegúrese de que el contexto está coloreando con negro (# 000).

Entonces se crea una pared- un hash que contiene un valor xy un valor de y.

Luego combinamos todo que para dibujar la pared en el lugar correcto utilizando fillRect. Los argumentos afillRect son el valor x del lado izquierdo, el valor y de la parte superior, la anchura, y la altura. llenamos los de, asegurándose de multiplicar por squareSize para los dos primeros con el fin de convertir los valores de la red para los valores de píxel.

Nótese que podemos acceder a los valores de un hash utilizando la tecla adecuada (la X o y clave en el pared). Esta es la forma más sencilla de acceder a los valores de un hash (aunque no es apropiado en cada situación, como veremos más adelante).

**Paredes, dispuestas ante nosotros**

Por supuesto, no queremos sólo una pared. Queremos un montón de paredes. Y, ya que no sólo va a estar atrayéndolo, vamos a querer ellos almacenados en el componente en lugar de crear en la función de dibujo.

Vamos a empezar simple y simplemente crear dos paredes.

41

|  |  |
| --- | --- |
| 6: Las barreras internas | 42 |

* paredes: [
* {X: 1, y: 1},

3{X: 8, y: 5}

4 ],

Esta propiedad está a sólo dos valores hash de pared dispuestos en una lista. El término de lujo para una lista como esta es unaformación. Estamos declarando la matriz poniendo una serie de cosas entre medias[ y ], Con cada elemento separado por una coma.

He aquí una simple matriz: dejar que la fruta = [ 'cereza', 'fresa', 'naranja'].

Nuestra gama de pared está un poco más complicado, ya que estamos creando una matriz de valores hash en lugar de cadenas. Ahora es más fácil de ver, dando a cada pared hash de su propia línea en lugar de enumerar todos juntos en una sola línea.

Ahora vamos a poner esa matriz de paredes de usar y dibujarlos.

**Dibujo todas las paredes**

Vamos a poner nuestro código en la función drawWalls:

* drawWalls: **función**() {
* **dejar** squareSize= **esta**.obtener('SquareSize');

3**dejar** CTX= **esta**.obtener('CTX');

4ctx.fillStyle = '# 000';

5

6**dejar** paredes= **esta**.obtener('paredes');

* walls.forEach (**función**(pared){
* ctx.fillRect (wall.x \* squareSize,

|  |  |
| --- | --- |
| 9 | criajo \* squareSize, |
| 10 | squareSize, |
| 11 | squareSize) |

1. })
2. },

Esto es muy similar a la primera vez que dibujamos una pared, pero con algunas diferencias clave. En primer lugar, en lugar de definir una pared singular dentro de la función, se obtiene la lista de las paredes del componente que hemos creado anteriormente. A continuación, se utiliza elpara cada método con el fin de sacar todas esas paredes.

Un poco más acerca de cómo la (muy bien llamado) para cada método funciona: Se toma una matriz, y luego se llama al para cadael método de la matriz, y luego le dan ese método una función. La función es lo que vas a hacer para cada elemento de la matriz.

Para nuestros propósitos, esa función se llevará a uno argumento, el elemento de la matriz. En nuestro código, que seríapared.

|  |  |
| --- | --- |
| 6: Las barreras internas | 43 |

Entonces, reconocerá la fillRectcódigo de nuestro ejemplo anterior. Vamos a estar usandofillRect para dibujar cada una de las paredes de la matriz.

**Llamar al Draw**

Ahora que tenemos el drawWallsmétodo, tenemos que llamarlo. Vamos a añadir justo antes de drawCircle en ambos lugares donde se encuentra drawCircle.

* didInsertElement () {
* **esta**.drawWalls ();
* **esta**.drawCircle ();4 }, 5

6 movePacMan (dirección, cantidad) {

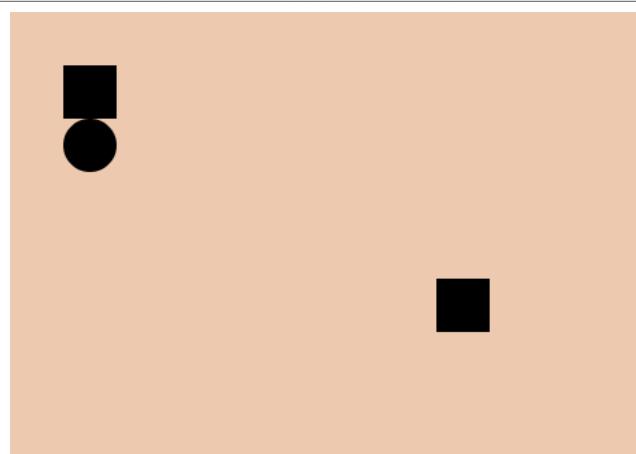
7**esta**.incrementProperty (dirección, cantidad);

8

9**Si**(**esta**.collidedWithBorder ()) {

1. **esta**.decrementProperty (dirección, cantidad)
2. }
3. **esta**.pantalla clara();
4. **esta**.drawWalls ();
5. **esta**.drawCircle ();
6. },

Esto nos lleva a las paredes presentan en nuestra red. Ahora es el momento de hacerlos cuentan.



**Las colisiones pared**

En el método movePacMan, tenemos este fragmento de código:

|  |  |
| --- | --- |
| 6: Las barreras internas | 44 |

* **Si**(**esta**.collidedWithBorder ()) {
* **esta**.decrementProperty (dirección, cantidad)

3 }

También queremos volver si chocamos con una pared. Vamos a hacer que esto suceda.

* **Si**(**esta**.collidedWithBorder ()|| **esta**.collidedWithWall ()) {

2**esta**.decrementProperty (dirección, cantidad)

3 }

No, no tenemos una collidedWithWall Método sin embargo ... estamos escribiendo cómo queremos usarlo en primer lugar, y luego vamos a hacer que el método.

Los 'métodos de llamada y luego definen ellos' estilo de programación puede ser muy liberador. En lugar de escribir todos los detalles de implementación en la salida, se puede escribir un conjunto de instrucciones claras y se preocupan por los detalles posteriores. Por supuesto, como todos los estilos de programación, es muy útil en algunas situaciones mientras que ser perjudiciales en otros.

los collidedWithWall método combina las técnicas lógicas booleanas que utilizamos en collidedWith-Frontera con un giro en el manejo de array que hemos utilizado en drawWalls.

* collidedWithWall: **función**() {

2**dejar** X= **esta**.obtener('X');

3**dejar** y= **esta**.obtener('Y');

4**dejar** paredes= **esta**.obtener('paredes');

5

6**regreso** walls.any (**función**(pared){

7**regreso** X==wall.x&&

8 y == criajo

* })

10 }

los algunamétodo devuelve un valor booleano. Pasa a través de la matriz y ejecuta una función en cada elemento.

Si el valor de retorno de cualquiera de ellos es cierto, entonces la alguna método devuelve verdadero.

La función utilizada en este alguna método está comprobando si el PAC es en la parte superior de una pared (The && símbolo significa 'y').

Por lo tanto, si el PAC es en la parte superior de las paredes, a continuación, collidedWithWalldevuelve verdadero. Ejecutar el código, trate de correr en una pared ... y funciona!

|  |  |
| --- | --- |
| 6: Las barreras internas | 45 |

**Resumen**

En este capítulo se utilizó el fillRectmétodo en nuestro contexto con el fin de trazar una pared. A continuación, hemos aprendido acerca de las matrices, y creamos una serie de paredes. Hemos creado eldrawWalls método que en bucle sobre esa matriz (utilizando para cada) Para dibujar cada pared. A continuación, creó lacollidedWithWall método que utiliza la algunaMétodo para comprobar si estábamos en la parte superior de las paredes. En el camino también hemos aprendido acerca de la lógica booleana y la|| (o y && (Y) los operadores lógicos.

Oí te gustó la parte acerca de las matrices, por lo que en el siguiente capítulo vamos a poner algunos arreglos en nuestras matrices. También estaremos comiendo nuestros primeros gránulos. ¡Emocionante!

**Despejando el camino**

“¿Cómo el entrenamiento ir hoy?”, Se pregunta Jerome.

Usted no quiere pensar en ello, mucho menos hablar de ello.

Después pequeña charla de Matteo acerca de su futura muerte, que había pasado unas horas sin incidentes navegar círculos cuadrados (?) Alrededor de una pared. Después de que lo hizo la figura de un ocho ejercicios. Fue sobre todo muy bien, pero a pesar de que lo mantenía unido y continuamente mejorado su tasa de choque, que estaba recibiendo cansado y los nervios empezaron a frazzle.

Luego vino la línea diagonal. Una serie de paredes en la que se vieron obligados a ir a la derecha, luego hacia abajo, luego a la derecha, luego hacia abajo, hasta el fondo de la línea. Un movimiento en falso, y que era el momento de un choque.

Usted ha hecho un montón de movimientos incorrectos.

Eventualmente se desplomó en el panel de control, apisonando en la misma pared una y otra vez mientras que el PAC formación que sorprendió por cada infracción. El último que recuerda es Matteo gritando, y luego algunas personas con camisas blancas que tirando de la PAC.

No hay manera de que usted está diciendo Jerome cómo el entrenamiento de hoy ha ido.

“Tienes tu primera visita enfermería?”

¿Es tan obvio? Jerome está siendo agradable porque te conocía antes de esto, cuando eras bueno en algo, pero sólo se puede imaginar lo que está pensando. Él sabe lo mal que se aspira. Si se lo puede ver, los demás pueden verlo? Y no van a ser tan agradable. Ellos simplemente se ignorarían. Al igual que están haciendo ahora. Y si era obvio que aspirado, Jerome sería agradable y disfrutar de la compañía de un viejo amigo mientras estaba todavía alrededor, que no sería por mucho tiempo. Y eso era lo que estaba haciendo ahora.

Oh Dios. Todos ellos saben que chupan.

Tal vez alguien les dijo. Tal vez las personas en las camisas blancas llevaron inconsciente a través de los pasillos y la gente vio. Tal vez todo el mundo vio. Al menos, todo el mundo parece saber, incluso si ellos no lo ven a sí mismos. ¿Por qué si no iban a ser estudiadamente haciendo caso omiso de ella? ¿Por qué si Jerome ser tan amable?

Jerome tose. “No se suponía que los choques para que se muda, simplemente inconsciente”.

Usted cerrarse de nuevo, y unas pocas palabras captura. Los choques se supone que esté inconsciente. Y él había dicho “primera” visita de enfermería, las personas que significan hizo varias visitas. Era normal para hacer visitas múltiples. ¿Derecha?

“¿Cómo sabe que fui a la enfermería?” Le pregunte.

Señala a su brazo. “quemaduras de choque. Esto ocurre cuando se contrae dentro de una formación PAC “.

46

|  |  |
| --- | --- |
| Despejando el camino | 47 |

“Entonces ... esto sucede mucho? Es decir, se supone que esto suceda?”

“Hemos construido la frustración y el dolor en el entrenamiento, porque cuando estás abajo en las minas de pellets que habrá un montón de frustración y dolor. Si usted consigue todo bien desde el primer intento, o si la práctica hasta que es perfecto, entonces todo lo que aprenderá en la forma de operar un PAC. Pero ser un Pac-Man es mucho más que eso. Está siendo sacudió pero mantener el control. Se trata de tener un día frustrante, tal vez incluso ir a la enfermería, y luego volver a estar arriba y tratando de nuevo mañana “.

“Pero no quiero que se sienten frustrados. Me siento como un fracaso en este momento. En la enfermería Vi videos de estos PAC-hombres que navegan las minas y hacen que parezca tan fácil. No creo que pueda llegar a ser tan bueno como estos chicos. Y si los chicos aquí alguna vez averiguan lo malo que soy ...”

"Mira. Todos hemos sentido esa manera. Puede ser más intimidante para usted ya que eres de un fondo diferente y no está seguro de si realmente perteneces aquí, pero todo el tiempo dedicado confundido, el aprendizaje, golpeando alrededor de las salas de formación. Ves todas las cicatrices en todo el mundo a tu alrededor? No consiguieron los de ser perfecto. Incluso después de la formación “.

“Pero tengo miedo de que si le digo a alguien aquí lo que pasó hoy, van a reír.”

“Sí, se reirán. Y entonces todo el mundo que no es un idiota compartirá una historia en la que perdió aún peor. Y luego vas a reír “.

“¿Estás diciendo que soy tan bueno como cualquiera aquí.”

Jerome se ríe. “Todavía no, pero yo te conozco, he visto tu mente, he visto trabajar. Si nos atenemos a ella podrás entre los mejores. Ahora deja que te enseñe cómo pensar como un Pac-Man “.

Él saca una hoja de papel y empieza a dibujar una cuadrícula ...

**7: The Grid**

**Al ver la cuadrícula**

En el capítulo anterior hemos creado algunas paredes que podríamos poner en el centro de nuestro campo de juego. Eso fue muy bien, pero la forma en que los especificamos hacía difícil visualizar:

* paredes: [
* {X: 1, y: 1},

3{X: 8, y: 5}

4 ]

¿Sabe lo que el campo de juego se verá así? Sólo puedo imaginar que si pienso realmente difícil. Agrega en diez paredes más y usted puede olvidarse de la comprensión de su nivel.

Vamos a reemplazar ese paredes matriz con una cuadrícula:

* *// 0 es un espacio en blanco*

2 *// 1 es una pared*

3 cuadrícula: [

4[0, 0, 0, 0, 0, 0, 0, 1],

5[0, 1, 0, 1, 0, 0, 0, 1],

6[0, 0, 1, 0, 0, 0, 0, 1],

7[0, 0, 0, 0, 0, 0, 0, 1],

8[0, 0, 0, 0, 0, 0, 0, 1],

9[1, 0, 0, 0, 0, 0, 0, 1],

1. ],

Como dice el comentario, '1' representa una pared y '0' representa un espacio en blanco. Ahora, a pesar de que tenemos diez paredes, es fácil de entender nuestro nivel.

La estructura que estamos utilizando para hacer esto es una matriz de matrices.

Aquí estamos optimizando para el aspecto visual de la red, de manera que se pueda analizar fácilmente mediante al espacio 2d. Los números son perfectas para esto.

Sin embargo, la desventaja es que '1' es una representación terrible para 'pared', y tenemos que utilizar un comentario para explicar esto. Si procesamos que '1' más adelante en el código, que no podrá contar con el contexto y que puede tropezar.

Hay técnicas para ganar tanto la claridad en la representación de la red y la claridad en el código en otro lugar, pero no van a ser cubiertos en este libro.

48

|  |  |
| --- | --- |
| 7: The Grid | 49 |

**Matrices de matrices**

el último capítulo que empezamos a utilizar matrices. Array es sólo un término de lujo para 'lista'. Es un montón de artículos en el medio[ y ], Y separados por comas.

Aquí está una matriz de cadenas: [ 'Cereza', 'fresa', 'naranja']

Aquí hay una serie de números: [0, 1, 0, 1, 0, 0, 0, 1]

Ese conjunto de números que pasa a representar una fila (ya que sabemos lo que 0 y 1 significa en este contexto).

Una matriz de matrices es sólo una lista de otras listas. En este caso, una lista de las filas de nuestra red.

**Viendo la cuadrícula**

Anteriormente, nuestra drawWalls función en bucle a través de la lista de paredes (usando para cada) Y señala a cada uno:

* drawWalls: **función**() {
* **dejar** squareSize= **esta**.obtener('SquareSize');

3**dejar** CTX= **esta**.obtener('CTX');

4ctx.fillStyle = '# 000';

5

6**dejar** paredes= **esta**.obtener('paredes');

* walls.forEach (**función**(pared){
* ctx.fillRect (wall.x \* squareSize,

|  |  |
| --- | --- |
| 9 | criajo \* squareSize, |
| 10 | squareSize, |
| 11 | squareSize) |

1. })
2. }

Con la representación rejilla, la drawWalls función de bucle a través de cada célula con un bucle anidado, y luego dibujar una pared si el valor es '1' (no se preocupe, vamos a explicar cada parte de este código):

|  |  |
| --- | --- |
| 7: The Grid | 50 |

* drawWalls: **función**() {
* **dejar** squareSize= **esta**.obtener('SquareSize');

3**dejar** CTX= **esta**.obtener('CTX');

4ctx.fillStyle = '# 000';

5

6**dejar** cuadrícula= **esta**.obtener('cuadrícula');

7grid.forEach (**función**(Fila, rowIndex) {

8row.forEach (**función**(Cell, columnIndex) {

9**Si**(celda== 1) {

1. ctx.fillRect (columnIndex \* squareSize,

|  |  |
| --- | --- |
| 11 | rowIndex \* squareSize, |
| 12 | squareSize, |
| 13 | squareSize) |

1. }
2. })
3. })
4. },

Así que primero vemos que aunque todavía usamos para cada, La función que le damos ahora toma un segundo argumento. Ese argumento es el índice de la matriz. La función podría haber tomado ese argumento último capítulo, pero no lo necesitamos así que lo dejamos fuera. Ahora que lo necesitamos, estamos poniéndolo de nuevo. Hay un tercer argumento disponible, que es toda la matriz de que lapara cada actúe por.

Si usted quiere tener el tercer argumento, que necesita los dos primeros. Si desea que el segundo argumento, la primera es necesaria, pero el tercero es opcional. Si todo lo que está utilizando es el primer argumento, no es necesario especificar cualquiera de los otros argumentos.

Así que estamos teniendo cada fila, y para cada fila que estamos buscando en cada celda (cada número en la matriz fila). Si el número es 1, entonces estamos dibujando un rectángulo. Anteriormente sabíamos dónde trazar el rectángulo con la valor X e Y de nuestra hash de la pared, pero ahora estamos utilizando los índices de la celda de la cuadrícula.

Una cosa a observar es que las matrices en javascript están basados ​​en cero, lo que significa que el índice para el primer elemento de la matriz es 0, y el índice para el segundo elemento de la matriz es 1, y el índice para el tercer elemento de la matriz es 2, y así sucesivamente. Esto funciona para nosotros, ya que estamos dibujar rectángulos desde el angular superior izquierda si queremos que la primera celda de la primera matriz para ser una pared, vamos a querer especificar [0,0] y no [40, 40 ] como la esquina superior izquierda.

Así es como nos mostramos la paredes-, pero se dará cuenta de que todavía estamos dibujando un espacio mucho más grande para el nivel. Vamos a arreglar eso.

Notamos que porque nos mantenían todos nuestros cambios con drawWalls, No hay que cambiar nada en los dos lugares en los que hemos llamado drawWalls. Tres hurras por la abstracción!

|  |  |
| --- | --- |
| 7: The Grid | 51 |



Si intenta reproducir en este momento, no va a funcionar. Esto se debe a que no hemos actualizado la detección de colisiones. Vamos a arreglar esto antes de que el capítulo se hace!

**El cálculo de la anchura y la altura del nivel**

En el capítulo 5 fijamos nuestra la anchura y la altura de nuestro elemento canvas como esto:

* <**lona** carné de identidad="myCanvas" anchura={{}} ScreenPixelWidth altura={{}} ScreenPixelHeight></ \

2 canvas>

Y luego se calculó el PixelWidth y pixelHeight como esto:

* PixelWidth: 20,

2 pixelHeight: 15,

3 screenPixelWidth: Ember.computed (**función**() {

4 **devolver este**.obtener('ScreenWidth') \* **esta**.obtener('SquareSize') 5 }),

6 screenPixelHeight: Ember.computed (**función**() {

7 **devolver este**.obtener('ScreenHeight') \* **esta**.obtener('SquareSize') 8 })

El problema aquí es que la anchura y la altura han cambiado. Ya no son simplemente declarada-están ahora sobre la base de la red. Así que vamos a utilizar la red para calcularlos. Vamos a empezar con la altura.

|  |  |
| --- | --- |
| 7: The Grid | 52 |

* ScreenHeight: Ember.computed (**función**() {

2**devolver este**.obtener('Grid.length');

3 }),

Esto está fijando la altura a la longitud de la red de rejilla, que es el número de filas que están en la rejilla. En la red que hemos creado anteriormente en este capítulo, que sería 6.

'Longitud' es una propiedad que se encuentra en cada matriz de JavaScript, y hace exactamente lo que cabría esperar.

**profunda se**

Tenga en cuenta que this.get ( 'grid.length') es lo mismo que this.get ( 'rejilla'). longitud. Ascuasobtenermétodo es capaz de propiedades de la cadena separándolas con un '' en la cadena.

La versión combinada tiene la ventaja, además de ser más fácil de leer, de no hacer estallar si la rejilla es nulo o no definido.

Haremos algo similar con el ancho:

* ScreenWidth: Ember.computed (**función**() {
* **devolver este**.obtener('Grid.firstObject.length')3 }),

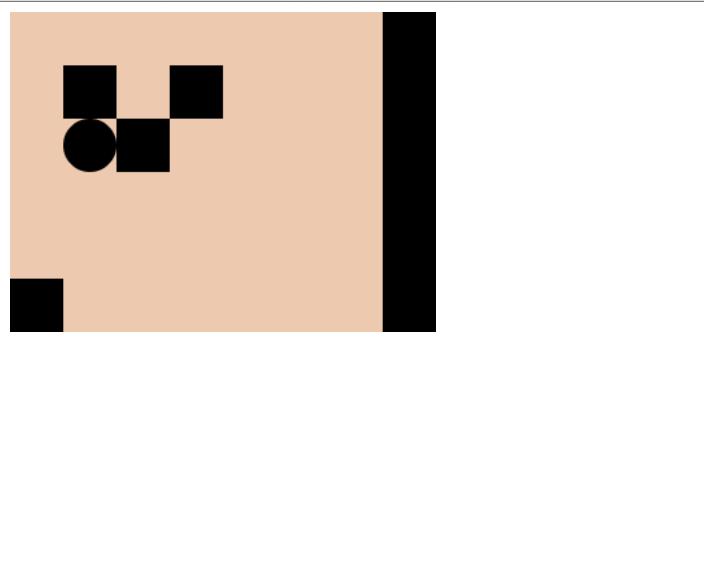
Esto toma la rejilla, agarra la primera fila (con firstObject), Y obtiene la longitud de esa primera fila (en este ejemplo: 8).

'FirstObject' es una propiedad que se encuentra en cada serie Ember (pero no toda matriz de JavaScript) que obtiene el primer elemento de la matriz.

Tenga en cuenta que this.get ( 'grid.firstObject.length') es lo mismo que this.get ( 'rejilla'). get ( 'firstObject'). l

¿Por qué utilizar una obtenerpara firstObject pero no la longitud? propiedades Ember deben ser recuperados con elobtenermétodo, no accede directamente. propiedades de JavaScript se puede acceder directamente, o pueden ser recuperados con elobtenerpalabra clave. Así que la última expresión también podría expresarse comothis.get ( 'rejilla'). get ( 'firstObject'). get ( 'longitud').

|  |  |
| --- | --- |
| 7: The Grid | 53 |



Ahora nuestro lienzo se ajusta a nuestra red, porque hemos calculado correctamente el ancho y la altura de la rejilla. Para finalizar este refactor, vamos a obtener nuestra PAC a moverse de nuevo.

**Mueve a través del espacio**

El movimiento dejó de funcionar razón es porque el collidedWithWallmétodo aún espera una serie de paredes. No se puede pedir una serie de paredes más, así que se detenga el programa continúe. Ahora vamos a cambiarlo a trabajar con nuestra red.

Recuerda el collidedWithWallmétodo comprueba para ver si nuestra PAC había dado en cualquiera de las paredes. He aquí cómo lo hicimos antes:

* collidedWithWall: **función**() {

2**dejar** X= **esta**.obtener('X');

3**dejar** y= **esta**.obtener('Y');

4**dejar** paredes= **esta**.obtener('paredes');

5

6**regreso** walls.any (**función**(pared){

7**regreso** X==wall.x&&

8 y == criajo

* })

10})

11 },

Así es como lo estamos haciendo ahora:

|  |  |
| --- | --- |
| 7: The Grid | 54 |

* collidedWithWall: **función**() {

2**dejar** X= **esta**.obtener('X');

3**dejar** y= **esta**.obtener('Y');

4**dejar** cuadrícula= **esta**.obtener('cuadrícula');

5

6**regreso** rejilla [y] [x]== 1

7 },

Antes, estábamos recorriendo la lista de las paredes y el control de uno contra uno. Ahora, debido a que estamos en un punto específico en el que la rejilla, podemos comprobar que acaba de rejilla.

Estamos comprobando ese punto de la red a través de descriptores de acceso de la matriz. Son un medio rápido para encontrar un artículo en un punto específico en una matriz. Así que aquí es cómo funcionaría con una matriz:

* **dejar** frutas=['Cereza','fresa','naranja']

2 frutos [0] *//'Cereza'*

3 frutos [2] *//'naranja'*

Recuerde, las matrices se indexan-cero, por lo [0] obtiene el primer artículo, y [2] obtiene el tercer punto.

Así que vamos a aplicar a una cuadrícula bidimensional simplificado:

* **dejar** cuadrícula=[
* [0, 0, 0, 0, 0],

3[0, 1, 0, 1, 0],

4[0, 0, 1, 0, 0],

5 [0, 0, 0, 0, 0] 6 ]

7 cuadrícula[0] *// [0, 0, 0, 0, 0]*

8 cuadrícula[1] *// [0, 1, 0, 1, 0]*

9 cuadrícula[1] [1] *// 1*

1. cuadrícula[1] [2] *// 0*

Así que el primer descriptor de acceso recibe la fila (el valor de y), y el segundo acceso obtiene la columna de esa fila (el valor de x).

Nuestro nuevo collidedWithWall función solo se encuentra el número que está en esa coordenada y devuelve verdadero si es igual a 1 (si se trata de un muro).

**Resumen**

Por lo que en este capítulo no construyó ninguna nueva funcionalidad en el juego ... pero cambió por completo nuestra forma de pensar acerca de cómo trabajar con los niveles en el juego, y lo hizo más fácil para nosotros entendemos.

|  |  |
| --- | --- |
| 7: The Grid | 55 |

Durante el transcurso de la refactor hemos aprendido sobre matrices de matrices, la matriz descriptores de acceso (a través de la siguiente sintaxis: [entero]), el longitud y firstObject propiedades de las matrices y los argumentos adicionales para la función que alimentamos para cada.

También establecemos las bases para la adición de algo pastillas-que habría sido extremadamente difícil de hacer sin el sistema de red que hemos creado este capítulo.

**primera cosecha**

La explicación de Jerome de la parrilla realmente helped- es como se puede ver toda la habitación de su mente ahora, como un mapa, ya dónde serán los pilares.

Ahora está perfectamente preparado para el entrenamiento de ayer.

“Está bien,” Matteo dice: “¿Estás listo para su primer día en las minas?” Cae la mandíbula. “Un día en las minas? ¿Ya?"

Matteo ríe. “No los grandes minas chico. La mina de entrenamiento. Ven."

Se puede caminar a una puerta diferente a la de hoy y tomar un ascensor. Una formación de PAC está rodando en la sala como llegues, y él hace un gesto que se ponga en.

Después de que esté en el PAC, sacas en lo Matteo está llamando la mina de entrenamiento. Las paredes no son tan suave -Han sido desconchados en lugares-y los pilares están hechos de tierra sólida y no son capaces de ir arriba y abajo como los que hay en la sala de entrenamiento.

También es considerablemente más oscuro, iluminado sólo por los fluorescentes débiles en el techo y el suave resplandor de una pastilla en granza.

“Estas pastillas, que son lo que nos da todo lo que disfrutamos en nuestra vida moderna, incluyendo nuestra comida, nuestro poder, nuestra ...”

“Yo sé”, se interrumpe. “He pasado toda mi vida viene con nuevas y más eficientes formas de utilización de los pellets.”

“Ah. Bueno. Sí. Es por eso que te trajeron, ¿no es así.”‘¿Qué me puede decir sobre eso?’

Hace una pausa, se detuvo en seco. Casi se puede ver la información de ser detenido por la seguridad ya que comienza y deja de hablar varias veces.

Otra voz se corta en, la de Terrance Mayhew. “Salam”, dice, “¿qué sabes de las implicaciones de su trabajo más reciente?”

Te paras a pensar ... se trataba de una nueva forma de radiación que proviene de pastillas especialmente densas, y las interacciones que tuvo con materiales orgánicos y metales. Nada especial, simplemente reflexiones teóricas sobre una forma rara de la materia.

“Se puede ... nos proporcionan una forma más eficiente de la energía?” Había sido un proyecto paralelo, uno que había pensado durante años, pero sólo recientemente se les ha dado los recursos para, por lo que nada grande había venido de él todavía. Ya era bastante temprano y esotérica suficiente como para que su laboratorio compañeros ni siquiera se habían molestado en averiguar lo que estaba trabajando.

56

|  |  |
| --- | --- |
| primera cosecha | 57 |

Terrance ríe. “Sí, esa es una manera de decirlo. No se le ha dado acceso a esta, pero de vuelta antes de que el califato, cuando Yihad era externa, no era un arma. Un arma de energía, un arma de energía muy eficiente. Podría nivelar toda una ciudad. Después de la Yihad final fue completa destruimos todo lo que pudimos encontrar, así que nunca se pueden usar de nuevo. Hemos destruido los planes que fueron utilizados para crearlos. Incluso ... bueno, después de habernos hecho, nadie en el mundo sabía cómo construir uno de ellos. Nosotros realmente pensamos que tendríamos no hay otros enemigos. Y durante siglos, esto era cierto “.

“Pero ahora han llegado a los fantasmas.”

“Sí, los fantasmas han llegado. Y necesitamos armas. Mal “.“Entonces, ¿por qué no deja que me quede en mi laboratorio? ¿Por qué me envía allí abajo?”

“Los sedimentos densos que estaba trabajando con ... esos eran sólo una muestra. Hay pastillas ultradensas allá abajo, en el fondo de las minas, que no sabemos cómo transportar aún. Vamos a conseguir que un PAC especial que puede manejar su equipo de laboratorio, para que pueda viajar allí y estudiarlos a sí mismo “.

Asientes. De repente tiene sentido, por la que se convocaron repente. Por eso mismo Terrance Mayhew está supervisando su formación. ¿Por qué le dieron fondos para el proyecto después de años de peticiones. Hay millones de pensamientos que giran alrededor de su cabeza, pero por ahora sólo tiene una pregunta para Terrance “Va a ser que me acompaña a las minas?”

"Sí. No puedo pretender comprender los detalles de su trabajo”(saltar internamente con orgullo),‘pero voy a estar allí para supervisar los detalles mecánicos.’

"¿Cuándo nos vamos?"

“Bueno, eso depende de usted y Matteo. Matteo?”

Matteo ha sido claramente estupefacto por las revelaciones que se acaban de compartir, pero que recupera su compostura rápidamente. Es un profesional.

“Debemos, uh, hay que trabajar en eso. Debe ser ... vamos a tratar de hacerlo rápido.”Él se dirige a ti, y mientras se empieza a hablar de la formación de su voz pierde el temblor y se calma y estable.

“Los pellets aquí, los pellets que has estado estudiando, aquí están en su estado en granza. Por lo general crecen en racimos más estrictas, pero esta mina se cosecha cada vez que un alumno nuevo llega a través.”Hace una pausa para recomponerse, y mira hacia arriba al bloque de oficinas donde Terrance debe ser. “Ahora conseguir recoger.”

**8: Pellets**

En el último capítulo refactorizamos el uso de sólo una serie de paredes para el uso de una rejilla. Si bien nuestro juego terminó buscando exactamente lo mismo después, estábamos en mejores condiciones para razonar acerca de cómo funcionaba el mundo del juego. En este capítulo vamos a tomar ventaja de eso refactor añadir pastillas para el mundo del juego.

Aquí está la parrilla después de lo llenamos con pastillas:

* *// 0 es un espacio en blanco*

2 *// 1 es una pared*

3 *// 2 es un pellet*

4 cuadrícula: [

5[2, 2, 2, 2, 2, 2, 2, 1],

6[2, 1, 2, 1, 2, 2, 2, 1],

7[2, 2, 1, 2, 2, 2, 2, 1],

8[2, 2, 2, 2, 2, 2, 2, 1],

9[2, 2, 2, 2, 2, 2, 2, 1],

1. [1, 2, 2, 2, 2, 2, 2, 1],
2. ],

Imagínese haciendo eso con sólo una serie de paredes y una serie de pastillas-sería muy difícil de imaginar, y podría llevar a errores como poner múltiples bolitas en la misma célula, o poner una pared y una bolita en la parte superior de uno al otro . Aquí vemos que no puede ser exactamente una cosa en cada célula, y (ahora) es ya sea una pared o una bolita.

Pero sólo tener en nuestra red no hace mucho. Tenemos que sacar las pastillas.

**dibujo Los pellets**

En el drawWalls método, se bucle a través de todas las células en la red a través de un bucle anidado, y luego, si una celda de la cuadrícula es una pared (== celular 1) Se dibuja un rectángulo.

Vamos a tomar ventaja de todo lo que la maquinaria que ya hemos construido y sólo tiene que añadir una pastilla de dibujo a la misma:

58

|  |  |
| --- | --- |
| 8: Pellets | 59 |

* drawWalls: **función**() {
* **dejar** squareSize= **esta**.obtener('SquareSize');

3**dejar** CTX= **esta**.obtener('CTX');

4ctx.fillStyle = '# 000';

5

6**dejar** cuadrícula= **esta**.obtener('cuadrícula');

7grid.forEach (**función**(Fila, rowIndex) {

8row.forEach (**función**(Cell, columnIndex) {

9**Si**(celda== 1) {

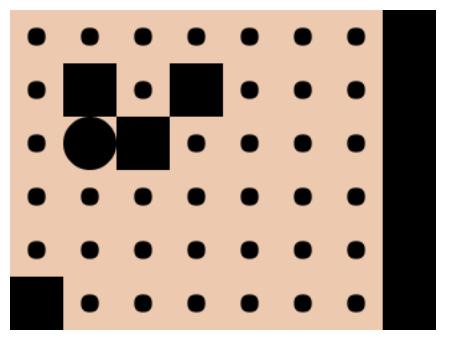
1. ctx.fillRect (columnIndex \* squareSize,

|  |  |
| --- | --- |
| 11 | rowIndex \* squareSize, |
| 12 | squareSize, |
| 13 | squareSize) |

1. }
2. *// cosas nuevas comienza aquí*
3. **Si**(celda== 2) {
4. **dejar** pixelX=(columnIndex+ 1/2)\*squareSize;
5. **dejar** pixelY=(rowIndex+ 1/2)\*squareSize;
6. ctx.beginPath ();
7. ctx.arc (pixelX, pixelY, squareSize/ 6, 0, Mates.PI \* 2, **falso**);
8. ctx.closePath ();
9. ctx.fill ();
10. }
11. })
12. })
13. },

Así que después comprobamos si la celda contiene una pared, después comprobamos si contiene una bolita. Si lo hace, se dibuja un círculo utilizando el código que se parece mucho a que utilizamos para llamar la PAC (pero más sobre esto más adelante).

|  |  |
| --- | --- |
| 8: Pellets | 60 |



**Di lo que haga, haga lo que dice**

Por supuesto, ahora drawWallsestá haciendo más que simplemente paredes de dibujo. Eso es confuso para los desarrolladores futuras (incluyendo futuro). Podríamos ampliarlo adrawWallsAndPellets, Pero creo que es un nombre mejor DibujarRejilla. Eso nos deja libres para añadir más cosas que sólo las paredes y pastillas sin necesidad de cambiar el nombre otra vez.

Así que cambiamos el nombre de la función a drawWalls DibujarRejilla, Y luego cambiarla en los dos lugares que se llama. Si no está seguro exactamente lo que quiero decir,[echa un vistazo a la modificación de la history⁹ git](https://github.com/jeffreybiles/chapter-by-chapter-game/commit/244f311bc0e04add2c36be49783a740bffd887f3).

Ahora DibujarRejillatiene el nombre correcto, pero todavía podemos hacerlo mejor. Hay dos masas de código, uno para la elaboración de una pared y uno para dibujar un pellet. Vamos a distinguir entre los usuarios y les dan nombres:

* drawWall: **función**(X, y) {
* **dejar** CTX= **esta**.obtener('CTX');
* **dejar** squareSize= **esta**.obtener('SquareSize');

4

5ctx.fillStyle = '# 000';

6ctx.fillRect (x \* squareSize,

|  |  |
| --- | --- |
| 7 | y \* squareSize, |
| 8 | squareSize, |
| 9 | squareSize) |

1. },
2. drawPellet (x, y) {
3. **dejar** CTX= **esta**.obtener('CTX')
4. **dejar** squareSize= **esta**.obtener('SquareSize');

⁹<https://github.com/jeffreybiles/chapter-by-chapter-game/commit/244f311bc0e04add2c36be49783a740bffd887f3>

|  |  |
| --- | --- |
| 8: Pellets | 61 |

15

1. **dejar** pixelX=(X+1/2)\*squareSize;
2. **dejar** pixelY=(Y+1/2)\*squareSize;
3. ctx.fillStyle = '# 000';
4. ctx.beginPath ();
5. ctx.arc (pixelX, pixelY, squareSize/ 6, 0, Mates.PI \* 2, **falso**);
6. ctx.closePath ();
7. ctx.fill ();
8. },
9. DibujarRejilla: **función**() {
10. **dejar** cuadrícula= **esta**.obtener('cuadrícula');
11. grid.forEach (**función**(Fila, rowIndex) {
12. row.forEach (**función**(Cell, columnIndex) {
13. **Si**(celda== 1) {
14. **esta**.drawWall (columnIndex, rowIndex);
15. }
16. **Si**(celda== 2) {
17. **esta**.drawPellet (columnIndex, rowIndex);
18. }
19. })
20. })
21. },

¡Esto es mucho mejor! DibujarRejilla se encarga de bucle a través de la red, y luego se llama adrawWall y drawPellet para dibujar cada entidad. Hemos cambiado el nombre de la larga columnIndex y rowIndex a X y y, Que será intuitivamente familiar a cualquiera que haya trabajado con gráficos. Incluso mejor, ahora está claro por el contexto que '1' significa pared y '2' significa pellets.

**Los problemas de alcance**

Pero no es todo sol y rosas (todavía). Si intenta ejecutar este código, se encontrará con el siguiente error:TypeError no detectada: No se puede leer la propiedad 'drawPellet' de la indefinida.

Esto sucede debido a cómo maneja Javascript alcance.

Cada vez antes, cuando llamamos esta, Hemos estado refiriendo a la PacMancomponente. Asi quethis.drawGrid () significa 'ir al componente, encontrar el método llamado DibujarRejilla, Y lo llaman'.

Sin embargo, en el código anterior del alcance de estaha cambiado. Se cambia debido a las funciones que se introduce en lapara cadafunción. No le hemos dado un nombre esas funciones, por lo que vamos a llamarlos '' funciones anónimas. Al crear una función anónima, se crea un nuevo ámbito de aplicación, por lo que el significado deestacambios. Y en este nuevo ámbito, no haydrawPellet función.

|  |  |
| --- | --- |
| 8: Pellets | 62 |

Volviendo a una vieja analogía, que es la diferencia entre decir “encontrar el sofá en esta casa” y “encontrar el sofá en esta cocina”. Estás en el ámbito mal!

**Las flechas de grasa**

Este cambio de alcance es terrible y es la causa de la gran cantidad de errores. Las personas que escriben Javascript han llegado con un montón de maneras ingeniosas para conseguir alrededor de él, pero, con mucho, la mejor es una nueva característica del lenguaje apodado 'flechas de grasa. Es básicamente una forma de declarar funciones anónimas que es más corta y no cambia el alcance.

* *// antigua declaración de la función anónima*

2 console.log (**esta**) *// foo*

3 myArray.forEach (**función**(ít){

4console.log (**esta**) *// bar*

5 })

6

7 *// declaración de la función nueva grasa flecha*

8 myArray.forEach ((punto)=>{

9console.log (**esta**) *// foo*

1. })

Así, utilizando la flecha de grasa (() =>) En lugar de la palabra clave function (función()) Conserva el alcance en lugar de crear una nueva.

Esta es la forma en la función DibujarRejilla ve ahora:

* DibujarRejilla: **función**() {
* **dejar** cuadrícula= **esta**.obtener('cuadrícula');
* grid.forEach ((fila, rowIndex)=>{
* row.forEach ((célula, columnIndex)=>{

5**Si**(celda== 1) {

6**esta**.drawWall (columnIndex, rowIndex);

* }
* **Si**(celda== 2) {
* **esta**.drawPellet (columnIndex, rowIndex);

1. }
2. })
3. })
4. })

Ahora estamos de vuelta a ver hermosas paredes y pellets en todas partes. Gracias, flecha grasa!

|  |  |
| --- | --- |
| 8: Pellets | 63 |

¿Por qué una flecha de grasa? ¿Qué es una flecha delgada?

Grandes preguntas!

Uno en un tiempo hubo una lengua llamada 'CoffeeScript'. Era un lenguaje que se parecía mucho a Javascript, pero era mucho más hermosa. Se mantuvo muchos de los conceptos básicos, mientras que lijar los bordes ásperos.

Uno de los bordes ásperos era declaración de la función. Tenía una flecha delgada '->' que funcionaba igual que la palabra clave function, pero era más corto. También tenía una flecha de grasa "*)*'Que conservan el alcance.

Cuando llegó el momento de actualizar javascript, los creadores se dieron cuenta de lo maravilloso que era coffeescript y robaron ya que muchos de sus mejores ideas que pudieron al mismo tiempo conservar la compatibilidad con versiones antiguas de JavaScript. Una de esas ideas fue la flecha de grasa, a pesar de que ahora es tristemente despojado de su hermano de la flecha delgada.

¿Todavía tiene problemas? Asegúrese de que ha cambiado tanto de las llamadas adrawWalls decir DibujarRejilla.

**Comparte tu círculo**

Mientras estamos en el tema de la refactorización nuestro código para que sea más claro, se puede notar que drawCircle y drawPellet mirar casi exactamente el mismo:

* drawPellet (x, y) {
* **dejar** CTX= **esta**.obtener('CTX')
* **dejar** squareSize= **esta**.obtener('SquareSize');

4

5**dejar** pixelX=(X+1/2)\*squareSize;

6**dejar** pixelY=(Y+1/2)\*squareSize;

7**dejar** radio=squareSize/ 6;

8

9ctx.fillStyle = '# 000';

1. ctx.beginPath ();
2. ctx.arc (pixelX, pixelY, radio, 0, Mates.PI \* 2, **falso**);
3. ctx.closePath ();
4. ctx.fill ();

|  |  |
| --- | --- |
| 8: Pellets | 64 |

1. },
2. drawCircle: **función**() {
3. **dejar** CTX= **esta**.obtener('CTX');
4. **dejar** squareSize= **esta**.obtener('SquareSize');
5. **dejar** X= **esta**.obtener('X');
6. **dejar** y= **esta**.obtener('Y');
7. **dejar** pixelX=(X+1/2)\*squareSize;
8. **dejar** pixelY=(Y+1/2)\*squareSize;
9. **dejar** radio=squareSize/ 2;
10. ctx.fillStyle = '# 000';
11. ctx.beginPath ();
12. ctx.arc (pixelX, pixelY, radio, 0, Mates.PI \* 2, **falso**);
13. ctx.closePath ();
14. ctx.fill ();
15. }

No sólo eso, sino que cuando decimos drawCircle, Lo que realmente queremos decir es drawPac. Así que vamos a cambiar el nombre que, a continuación, hacer que por lo que estas dos funciones puede compartir algo de código.

* drawPac () {
* **dejar** X= **esta**.obtener('X');

3**dejar** y= **esta**.obtener('Y');

4**dejar** radiusDivisor= 2;

5**esta**.drawCircle (x, y, radiusDivisor)

6 },

7

8 drawPellet (x, y) {

9**dejar** radiusDivisor= 6;

1. **esta**.drawCircle (x, y, radiusDivisor)
2. },

12

1. drawCircle (x, y, radiusDivisor) {
2. **dejar** CTX= **esta**.obtener('CTX')
3. **dejar** squareSize= **esta**.obtener('SquareSize');
4. **dejar** pixelX=(X+1/2)\*squareSize;
5. **dejar** pixelY=(Y+1/2)\*squareSize;
6. ctx.fillStyle = '# 000';

|  |  |
| --- | --- |
| 8: Pellets | sesenta y cinco |

1. ctx.beginPath ();
2. ctx.arc (pixelX, pixelY, squareSize/radiusDivisor, 0, Mates.PI \* 2, **falso**);
3. ctx.closePath ();
4. ctx.fill ();
5. },

Como se puede ver, era sólo una diferencia entre unas líneas drawPac y drawPellet. Ahora sólo tenemos que volver a donde estábamos dibujando con el PACdrawCircle y llama drawPac en lugar.

**El procesamiento de los gránulos**

Por lo que hemos pasado un montón de dibujar las pastillas y luego hacer nuestro código mejor momento, pero la PAC va a tener que procesar los gránulos. Vamos a empezar que aquí, y luego terminar en el siguiente capítulo. Nuestro objetivo ahora es hacer que los pellets desaparecen cuando el PAC visita su celular.

Vamos a empezar con averiguar dónde queremos colocar este código. Debemos comprobar que no existen pastillas cada vez que el PAC se traslada a una nueva plaza, y luego procesar el pellet si existe. Así que vamos a ponerlo en elmovePacMan método.

* movePacMan (dirección, cantidad) {
* **esta**.incrementProperty (dirección, cantidad);

3

4**Si**(**esta**.collidedWithBorder ()|| **esta**.collidedWithWall ()) {

5**esta**.decrementProperty (dirección, cantidad)

* }

7

* **esta**.processAnyPellets ()

9

1. **esta**.pantalla clara();
2. **esta**.drawGrid ();
3. **esta**.drawPac ();
4. },

Por eso, pusimos la llamada a processAnyPellets después de moverse y la comprobación de colisiones con las paredes, pero antes de volver a dibujar todo.

Esto es lo que la processAnyPellets la función se ve así:

|  |  |
| --- | --- |
| 8: Pellets | 66 |

* processAnyPellets: **función**() {

2**dejar** X= **esta**.obtener('X');

3**dejar** y= **esta**.obtener('Y');

4**dejar** cuadrícula= **esta**.obtener('cuadrícula');

5

6**Si**(GRID [y] [x]== 2) {

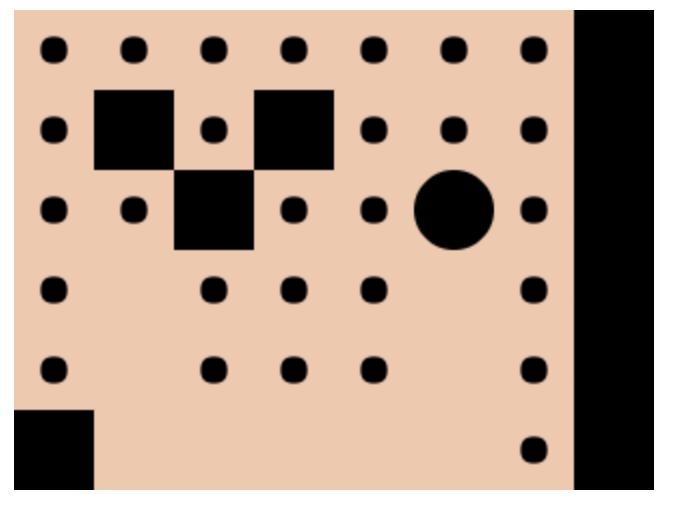
7rejilla [y] [x] = 0;

* }

9 },

Se comprueba para ver si la célula es igual a '2' (para ver si es un pellet), y si es entonces establece la célula a '0' (un espacio en blanco). Esto reemplazará el '2' que estaba en esa celda de la rejilla con un '0'. Es una misión, al igual que podríamos asignar a la variable 'x', o 'rejilla' 'y', pero está alcanzando en las matrices anidadas para realizar la tarea.

Ahora ejecuta el código y ver los pellets desaparecen debajo de su PAC!



Pero antes de seguir adelante, vamos a notar dos cosas acerca de este código. La primera es que '==' y '=' son muy similares, a pesar de que hacen cosas muy diferentes. Esto es algo que tendrá que permanecer en el puesto de observación para. Un error común es mezclar accidentalmente, y para asignar un valor a una variable cuando se quería comprobar para ver si era igual a la variable, o para regresar un valor lógico cuando se quería realizar una cesión.

La segunda cosa que puede notar es las similitudes con el collidedWithWall función:

|  |  |
| --- | --- |
| 8: Pellets | 67 |

* collidedWithWall: **función**() {

2**dejar** X= **esta**.obtener('X');

3**dejar** y= **esta**.obtener('Y');

4**dejar** cuadrícula= **esta**.obtener('cuadrícula');

5

6**regreso** rejilla [y] [x]== 1

7 },

Pero en este caso no vamos a refactorizar, porque a pesar de que son superficialmente similares que están sirviendo a diferentes propósitos y la combinación de ellos haría que el código sea más difícil de leer, no menos. Aunque por lo general es una buena idea para refactorizar código de distancia repetida, haciéndolo demasiado celo también puede conducir a problemas.

**Resumen**

En este capítulo hemos añadido a nuestra red de pastillas, las dibujó, refactorizamos nuestro código por lo que se denomina mejor y se repite menos, y empezamos a procesar las pastillas (al hacer desaparecer). En el camino hemos aprendido sobre las flechas de grasa, el alcance, las tareas en las matrices anidadas, y practicamos nuestras habilidades de refactorización.

En el siguiente capítulo finalizar el procesamiento de los pellets, que nos da una puntuación. También decidimos qué hacer cuando hemos terminado de procesar todos los gránulos.

**Ganancias y pérdidas**

Todavía se está moviendo un poco de las revelaciones, pero ahora hay que FOCUS- no puede ir a estrellarse contra las paredes ahora sólo porque eres importante para un plan maestro para salvar a la humanidad.

Matteo hace su parte para mantener las cosas normales, dando conferencias sobre los aspectos económicos de la recogida de gránulos.

“Cada sedimento representa una cierta cantidad de energía que se está viendo que en su contador de puntuación. Esa energía se utiliza para alimentar a nuestra gente, construir nuevos túneles, y en general a mantener el Califato corriendo.”Es opinión, esto es, literalmente, lo que solía pasar todo el día estudiando, pero se le dejó ir adelante. Es mantener su mente de todo lo demás.

Pero entonces se entra en territorio que no estás tan familiarizado. “Su PAC necesita energía también. En general, cuanto más abajo vas, más energía de la PAC takes- tanto, ya que tiene más abajo para ir, y también porque los líos atmósfera con nuestro procesamiento de pellets interna “.

Esa explicación no suena del todo bien, pero obviamente es suficiente para que los PAC-hombres. Haces una nota mental para aprender más sobre el fenómeno una vez que esto ha terminado. ¡Ver! Ya estás normalizar las cosas. Matteo sigue hablando. “El PAC de entrenamiento es que estás en uno de los modelos menos eficientes, pero eso es bueno- que lo utiliza solamente en los niveles superiores, y sólo durante el entrenamiento. Como es, ejecutarlo en este nivel se trata de romper incluso en términos de consumo de energía “.

Sorprendentemente, no se han dado en ningún paredes todavía- que está cada vez mejor! -y luego de recoger el sedimento final. Su pantalla se oscurece y se oye un ruido mecánico nuevo, uno distinto de los ruidos de la PAC. maquinaria más grande.

"¿Qué esta pasando?"

“Ah sí, eso es ... deberíamos haber convertido esa función. Si no se detectan pastillas de cercanías, la PAC pasa automáticamente al siguiente nivel. Es increíblemente eficiente, en términos de pastillas por minuto. Poner en los ascensores era caro, pero han más que pagado por sí mismos “.

Usted distraídamente recoger bolitas mientras habla, y te das cuenta: esto podría ser divertido. Esto sin duda podría ser divertido.

68

**9: Marcadores y Niveles**

**Manteniendo el marcador**

En el último capítulo empezamos el procesamiento de las pastillas, pero en realidad todo lo que hicimos fue hacer que desaparezcan. En este capítulo, vamos a dar a los gránulos de un propósito más elevado. Y que el propósito superior lleva el marcador.

Vamos a empezar ser la adición de una propiedad puntuación al componente PacMan:

* **por defecto de exportación** Ember.Component.extend (KeyboardShortcuts, {

2Puntuación: 0,

* ...

4 });

A continuación, vamos a mostrar que la puntuación en la plantilla de manillar para ese componente, justo debajo del campo de juego:

* *<! - templates / componentes / PAC-man.hbs ->*
* <**lona** carné de identidad="myCanvas" anchura={{}} PixelWidth altura={{}} PixelHeight> </**lona**>

3 <**br**>

4 Puntuación: {{score}}

Recuerda, {{Puntuación}}es el seguimiento de la propiedad puntuación en el componente pac-man. Así que ahora mismo se muestra una puntuación de 0, y se seguirá mostrando dicho puntaje, sin importar dónde va nuestra PAC, hasta que escribir código que hace que cambie. Por suerte, eso es relativamente fácil:

* processAnyPellets: **función**() {

2**dejar** X= **esta**.obtener('X');

3**dejar** y= **esta**.obtener('Y');

4**dejar** cuadrícula= **esta**.obtener('cuadrícula');

5

6**Si**(GRID [y] [x]== 2) {

7rejilla [y] [x] = 0;

* **esta**.incrementProperty ('Puntuación')

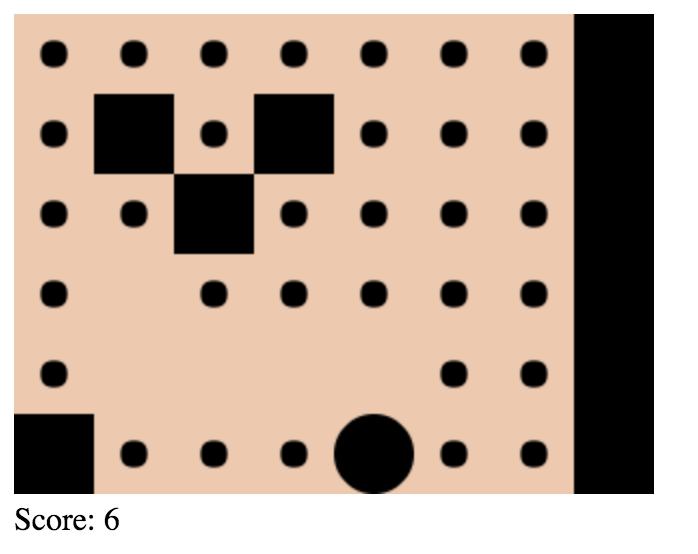
9}

1. },

69

|  |  |
| --- | --- |
| 9: Marcadores y Niveles | 70 |

Acabamos de añadir la línea que se incrementa la puntuación después de la línea en la que hemos eliminado el pellet. Iniciar la reproducción y la puntuación va a subir! Una de las cosas impresionantes sobre Ember es que las propiedades de la plantilla cambian automáticamente cada vez que se cambian a otra parte. Eso se llama 'unión', y se usa mucho en las aplicaciones tradicionales Ember.



Así que fue relativamente fácil. Vamos a hacer algo más difícil.

**Elevar a mismo nivel**

La parte inicial de crear el sistema de nivel es muy similar.

* **dejar** PacMan=Ember.Component.extend (KeyboardShortcuts, {
* levelNumber: 1,
* ...

4 });

* <**lona** carné de identidad="myCanvas" anchura={{}} PixelWidth altura={{}} PixelHeight> </**lona**>

2 <**br**>

3 Puntuación: {{score}} **& Nbsp; & Nbsp; & Nbsp;** Nivel: {{}} levelNumber

Comenzamos en el nivel 1 y el nivel se muestra después de la partitura. & Nbsp;es el código HTML para el 'espacio de no separación'. Es una manera de conseguir algo de espacio entre 'Score' y 'Nivel', ya que múltiples espacios lisos en colapso html abajo a un espacio.

A continuación tenemos que decidir dónde incrementar el nivel. Incrementando el marcador era de fácil ya estábamos haciendo algo cada vez que nos encontramos en una pastilla, y así podríamos unir la acción de la subasta puntuación a ese trozo de código. Para hacer frente a los niveles que tendremos que crear un código que detecta cuando el nivel se ha completado.

Nuestros criterios para un nivel que se realiza es que el jugador ha tomado todas las pastillas. Aquí está el código:

|  |  |
| --- | --- |
| 9: Marcadores y Niveles | 71 |

* Nivel completado: **función**() {
* **dejar** hasPelletsLeft= **falso**;

3**dejar** cuadrícula= **esta**.obtener('cuadrícula');

4

* grid.forEach ((fila)=>{
* row.forEach ((celular)=>{

7**Si**(celda== 2) {

8hasPelletsLeft = **cierto**

* }

1. })
2. })
3. **regreso** !hasPelletsLeft;
4. },

Esto es lo que está pasando en esta función:

1. Hemos establecido la primera hasPelletsLeft false.
2. Nos bucle a través de cada celda de la cuadrícula. Si alguno de ellos es igual a '2' (si tienen un pellet) A continuación, establecemoshasPelletsLeft true.
3. Volvemos al contrario de hasPelletsLeft (!significa 'no', por lo que se convertiría en una verdadera una falsa, y viceversa). Así que si se detecta un sedimento yhasPelletsLeft era verdad, Nivel completadovolvería falsa. Si no se detectaron un sedimento y sehasPelletsLeft quedado falsa, Nivel completado volvería realidad.

Así que eso es lo que usamos para determinar si el nivel se ha completado. Podemos comprobar que los criterios cada vez que el PAC recoge un pellet:

* processAnyPellets: **función**() {

2**dejar** X= **esta**.obtener('X');

3**dejar** y= **esta**.obtener('Y');

4**dejar** cuadrícula= **esta**.obtener('cuadrícula');

5

6**Si**(GRID [y] [x]== 2) {

7rejilla [y] [x] = 0;

* **esta**.incrementProperty ('Puntuación')

9

1. **Si**(**esta**.Nivel completado()){
2. **esta**.incrementProperty ('LevelNumber')
3. **esta**.volver a empersar el nivel()
4. }
5. }
6. },

|  |  |
| --- | --- |
| 9: Marcadores y Niveles | 72 |

Después de una pastilla se recogió y la puntuación se incrementa, tenemos entonces a comprobar si era el último (si el nivel se ha completado). Si es así, entonces incrementamos lalevelNumber propiedad y luego reiniciar el nivel.

Esto es lo que reiniciar el nivel se ve así:

* volver a empersar el nivel: **función**() {

2**esta**.conjunto('X',0);

3**esta**.conjunto('Y',0);

4

5**dejar** cuadrícula= **esta**.obtener('cuadrícula');

6grid.forEach ((fila, rowIndex)=>{

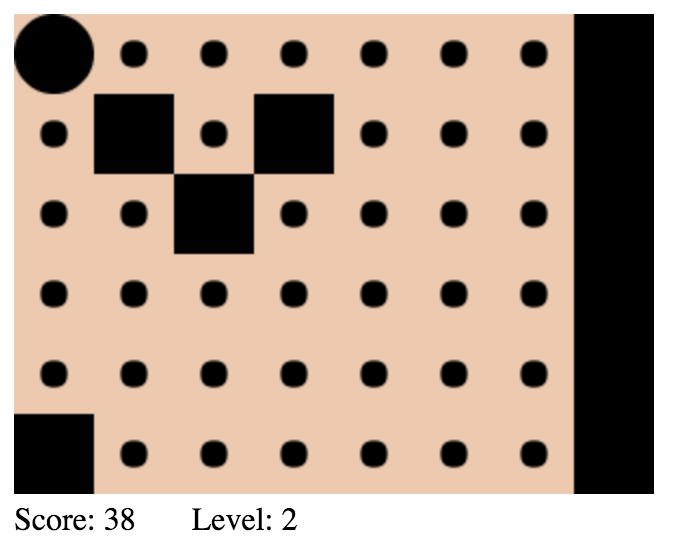
7row.forEach ((célula, columnIndex)=>{

8**Si**(celda== 0) {

9rejilla [rowIndex] [columnIndex] = 2

1. }
2. })
3. })
4. },

En primer lugar nos movemos a la PAC [0,0] (La izquierda lejos superior), a continuación, vamos a través de cada celda vacía en la red y poner una pastilla en el mismo.



**nueva sintaxis**

Recuerde que dijimos que había una mejor manera de declarar funciones? Estamos introduciendo ahora, porque quiero empezar a escribir con él. La buena noticia es que es bastante simple y que hará que nuestro código se ven más bonitas.

Así que vamos a decir que tenía una función como esta:

|  |  |
| --- | --- |
| 9: Marcadores y Niveles | 73 |

* addTwo: **función**(Num) {

2**regreso** num+ 2;

3 }

Bastante fresco ... pero podemos hacerlo más conciso:

* addTwo (num) {

2 **regreso** num+ 2

* }

Ahora que usted sabe que nombre(){} es una forma válida de declarar una función, y de hecho es el mismo que Nombre: function () {}, Se puede ver que los dos puntos y la palabra funciónen realidad no añadir mucha información. Ellos estaban muy bien por un tiempo, para recordar que se trata de una función que estamos creando, pero ahora no las necesitan.

Vamos a hacer que esta sintaxis cambiar todas partes que declaramos una función. Grabación de todos los que están aquí en el libro sería tonto, así que sólo vamos a enlazar a[el diff github donde hicimos](https://github.com/jeffreybiles/chapter-by-chapter-game/commit/deac876c39833845bcb7ef905f913e9d6307b07f) [todas esas changes¹⁰](https://github.com/jeffreybiles/chapter-by-chapter-game/commit/deac876c39833845bcb7ef905f913e9d6307b07f).

Si está utilizando un editor de texto con una buena búsqueda y reemplazo característica, intente buscar ':

función (' y la sustitución de todos los casos con '('.

**Resumen**

En este capítulo se revisaron una gran cantidad de conceptos, especialmente el 'ciclo a través de la red' patrón. Hemos aprendido algunas cosas menores, como el booleano!, Encuadernaciones manillar, y una nueva declaración de la función, pero sobre todo esto fue un respiro para solidificar lo que ya sabemos y obtener algunas características interesantes en el juego.

Ahora es el momento de cinturón de seguridad, debido a que los próximos capítulos van a ser un extraordinario.

¹⁰<https://github.com/jeffreybiles/chapter-by-chapter-game/commit/deac876c39833845bcb7ef905f913e9d6307b07f>

**Suave, eficiente**

“He oído rumores de que los fantasmas son cada vez más cerca.” Jerome levanta la vista de su plato.

“Dicen que la resistencia en Asia se ha ido.”

Se encoge de hombros. “Eso no es realmente nuestro problema.” Pero fue un encogimiento de hombros tensos, practicado. Todo en él era tensa. “Vamos a hablar de su entrenamiento.”

“He escuchado la charla desde que llegó aquí. Después de que terminen con Asia, no hay nada que impida venir a California “.

Sus ojos se estrecharon mientras él trató de parecer relajado. No estaba funcionando. “Nunca hubo nada que les impida venir a California. Son extranjeros. Tienen naves espaciales. Si quisieran estar aquí, estarían aquí “. Se supone que es reconfortante? “Ahora, vamos a hablar de su entrenamiento.”

Suspiras. “Está bien, vamos a hablar de mi entrenamiento.”

“¿Cómo está usted los niveles de estrés?”

Te ries. Su mandíbula se apretó y el puño es agarrar el tenedor como que va a huir y comenzar una yihad. “¿Cómo están sus niveles de estrés?”

Hace una pausa y se ríe demasiado bruscamente. Su actitud cambia por completo. “Lo sentimos, no ha habido un poco de materia en el laboratorio, es que ... quería asegurarse de que estaba haciendo bien. Que su cuerpo estaba sosteniendo bajo el nuevo estrés “.

"Estoy bien. No hay nuevos moretones “.

"Bueno. Estaba preocupado porque están a punto de empezar en el siguiente nivel de capacitación en APA.”‘¿En serio?’

Hace una pausa. "O tal vez no. Eso es lo que suelen hacer. Es decir, la secuencia es, a estas horas a la formación, que se inicia en una nueva “.

“Dicen que mi entrenamiento va más rápido de lo normal.”

“Sí, bueno, es ... de todos modos, me alegro de que estés mejor. Tal vez se ralentizó debido a las contusiones que tenía antes “.

¿Por qué es tan raro Jerome siendo hoy en día?

“Hábleme de su estrés en el trabajo”, dice usted.

“Ah, es ... ya sabes, fin del mundo y todo.” Ha vuelto a su estado normal, en broma su camino a través de desastres. “Estamos trabajando en una nueva variación de la PAC, uno diseñado para manejar equipos de la ciencia. Reconozco algunas de las ranuras “.

74

|  |  |
| --- | --- |
| Suave, eficiente | 75 |

Se piensa en decirle que es para usted, pero usted se abstenga. Usted no estaba seguro de lo que pasaba por ser un secreto militar por aquí, y es evidente que no le había dicho.

“De todas formas, tengo que irme. Y tienes que entrenar “.

Asientes y tomar su licencia. toda la caminata a las instalaciones de entrenamiento que usted está atormentado por alguna extraña sensación de que algo estaba pasando. Algo más de las conspiraciones ya estás al tanto.

Matteo te saluda, y se ve Terrance saludando desde una sala acristalada llena de monitores. “Le estamos moviendo hasta hoy”, dice Matteo. “El siguiente nivel de capacitación en APA.”

Su explicación es rápido-fuego de la nueva PAC se mueve suavemente de casilla en casilla en lugar de saltar, y mientras que hace más difícil el control, sino que también hace que sea mucho más eficiente en cuanto a energía.

Jerome había sido derecha que se estaban moviendo hasta un diferente PAC. ¿Cómo había sido tan precisa, tan precisa, en su predicción? Tal vez fue sólo una coincidencia.

De saltar, comenzar a recoger bolitas, y deja que tu mente se queda en blanco.

**10: Animaciones**

Por lo tanto, este es el capítulo donde descubrí que Pac-Man podría no ser el mejor juego de utilizar para la enseñanza de los principiantes ... Pero bueno, la programación de la vida real a menudo tienen que saltar fuera de su profundidad y luego tratar de nadar hasta la orilla antes de ahogarse . Así que esta es una gran preparación.

En la programación de la vida real, también hay tiburones.

La primera parte de este capítulo será otro “cosas reorganizar y luego todo se juega el mismo” sección ( “Refactoring”). La peor parte es por eso por lo que estamos haciendo los cambios no tendrá mucho sentido hasta llegar a la segunda parte de este capítulo. La buena noticia es que usted aprenderá un poco de la nueva sintaxis Javascript moderna, y llegar a ver los hashes utilizados en formas más complejas.

Al final de este capítulo nuestra PAC será deslizándose suavemente entre las plazas, en lugar de teletransportarse de plaza en plaza y ocupando toda la energía de pellets.

**coordinar llegar**

Vamos a empezar con la misteriosa pieza central de la refactor:

* direcciones: {
* 'arriba': {X: 0, y: -1},

3'abajo': {X: 0, y: 1},

4'izquierda': {X: -1, y: 0},

5'derecho': {X: 1, y: 0},

6 'detenido': {X: 0, y: 0} 7 },

Se trata de un hash que contiene otros 5 hashes- uno para cada uno arriba, abajo, izquierda, derechoy detenido. Cada uno de esos valores hash contiene exactamente dos claves:X y y. Codifican lo que significa cada dirección en términos de coordenadas. Es algo bastante básico, pero la forma en que la información se almacena puede parecer un poco diferente.

A continuación cambiamos lo que estamos pasando a movePacMan en atajos de teclado.

76

|  |  |
| --- | --- |
| 10: Animaciones | 77 |

* atajos de teclado: {
* arriba() { **esta**.movePacMan ('arriba');},
* abajo() { **esta**.movePacMan ('abajo');},

4izquierda() { **esta**.movePacMan ('izquierda');},

5derecho() { **esta**.movePacMan ('derecho');},

6 },

Anteriormente, estábamos pasando una coordenada (como X) Y luego un valor (ya sea 1 o -1). Ahora sólo nos queda pasar una cadena que representa una dirección. Que es fácil adivinar dónde va esto.

Vamos a tomar una descripción de alto nivel de la nueva movePacMan función, y luego vamos a profundizar en las nuevas funciones que estamos utilizando en su interior.

* movePacMan (dirección) {
* **Si**(!**esta**.pathBlockedInDirection (dirección)) {
* **esta**.conjunto('X', **esta**.nextCoordinate ('X', Dirección));

4 **esta**.conjunto('Y', **esta**.nextCoordinate ('Y', Dirección)); 5

* **esta**.processAnyPellets ();
* }

8

9**esta**.pantalla clara();

1. **esta**.drawGrid ();
2. **esta**.drawPac ();
3. }

Los últimos tres comandos son familiares, pero la inicial Sideclaración es todo nuevo. Se dará cuenta de que ya no Verificar la presenciacollidedWithBorder o collidedWithWall... podemos eliminar de forma segura esas funciones. Sustitución tanto de ellos espathBlockedInDirection.

Si la ruta no está bloqueado (nótese la ! dentro de Si declaración), nos propusimos la coordenadas X e Y a sus valores 'siguiente', basado en el nextCoordinatefunción. Luego de procesar las pastillas allí.

Tenemos dos nuevas funciones que tratar. Vamos a hacer frentenextCoordinate En primer lugar, porque es más sencillo y nos dará un buen primer vistazo a cómo estamos usando la direcciones picadillo.

**nextCoordinate e interpolada Cuerdas**

|  |  |
| --- | --- |
| 10: Animaciones | 78 |

* nextCoordinate (coordenadas, dirección) {
* **devolver este**.get (coordenadas)+ **esta**.obtener(`direcciones.**ps**dirección**}**.**ps**coordinar**}**`\3 ); 4 },

Estamos siendo aprobado una coordinar ( 'X' o 'y') y una dirección( 'Arriba', 'abajo', etc.). En el cuerpo de la función que estamos añadiendo los resultados de dos expresiones. La primera es la corrienteX o yvalor. El segundo está cavando profundamente en eldirecciones picadillo.

Este es un nuevo la sintaxis de la cadena interpolada. Si eldirección es 'arriba' o la coordinar Se 'y', entonces nuestra segunda expresión va a terminar como this.get ( 'directions.up.y') y resolver a -1. Si era 'correcto' e 'y', entonces nuestra segunda expresión haría en comothis.get ( 'directions.right.y') y resolver a 0.

¿Teniendo problemas? Asegúrese de utilizar el carácter 'back-tick' para la interpolación de cadenas, no el carácter de comilla simple. En la mayoría de los teclados es justo debajo de la tecla de escape.

Si usted no entiende cómo las diferentes variantes en la determinación de hash como lo hacen, miran hacia atrás al capítulo 7, cuando definimos ScreenWidth y ScreenHeight. Aquí, como allí, cavamos en varias capas con una.obtener por tener una cadena con múltiples .s. El diferente es ahora estamos definiendo algunos de los enlaces sobre la marcha.

Si le ayuda, también podríamos haber expresado nextCoordinate como:

* nextCoordinate (coordenadas, dirección) {
* **devolver este**.get (coordenadas)+ **esta**.obtener('direcciones') .get (dirección) .get (coordi\3 nate);

4 },

En otras palabras, esta función toma el valor actual de la X o y coordinar, a continuación, utiliza el direccioneshash y la dirección indicada para averiguar si y cuánto valor que tiene que cambiar. Se devuelve el nuevo valor.

En nuestros usos dentro movePacMan, Fijamos el X o y coordinarse con el valor devuelto.

**Mirando hacia el futuro con la cuadrícula**

Ahora tenemos que averiguar si nuestro camino está bloqueado o no. Haremos que usando nuestradirecciones hash y nuestra cuadrícula matriz de matrices.

Usted recordará nuestra red, se muestra en una forma muy simplificada aquí:

|  |  |
| --- | --- |
| 10: Animaciones | 79 |

* cuadrícula: [
* [1, 2, 2],

3[2, 2, 1],

4[1, 2, 1]

5 ]

1 representa una pared, mientras 2 representa un pellet (0 representa un espacio en blanco). Lo que vamos a hacemos, al igual que hicimos antes con collidedWithWall y collidedWithBorder, Es proyectar hacia el futuro y ver si estamos en un buen lugar o un mal lugar.

* pathBlockedInDirection (dirección) {
* **dejar** cellTypeInDirection= **esta**.cellTypeInDirection (dirección);
* **regreso** Ember.isEmpty (cellTypeInDirection)||cellTypeInDirection=== 1;4 }, 5

6 cellTypeInDirection (dirección) {

7**dejar** nextX= **esta**.nextCoordinate ('X', Dirección);

8**dejar** nextY= **esta**.nextCoordinate ('Y', Dirección);

9

1. **devolver este**.obtener(`Cuadrícula.**ps**nextY**}**.**ps**nextX**}**`);
2. },

Así que primero se corre cellTypeInDirection. Que utiliza el métodonextCoordinate, Que simplemente se explicó en la sección anterior, para obtener el siguiente X y yvalores. Entonces, estamos utilizando la interpolación de cadenas para encontrar esa posición en la parrilla.

Una cosa interesante a destacar es que es posible dar valores x e y que terminan siendo fuera de la red. Eso es fino que sólo volveráindefinido. Entonces esoindefinidoserá capturado por

Ember.isEmpty (cellTypeInDirection) antes, comprobamos que cellTypeInDirection === 1.

El resultado final es que si el siguiente celda está fuera de límites (indefinido) O una pared (1), Entonces volvemos cierto de pathBlockedInDirection.

**animaciones de alta calidad**

Los cambios en este capítulo hasta ahora sólo han sido configurado; que terminan haciendo exactamente lo mismo que antes. En esta segunda mitad del capítulo vamos a finalmente hacer el cambio de salto entrecortado de un rollo smooooooth animada.

Más específicamente, cuando llegamos a una dirección, el PAC comenzará a rodar en esa dirección. Una vez que se ha alcanzado la nueva plaza, se detendrá. A pesar de que está rodando, es no tomará ninguna entrada, pero una vez que se detuvo se puede mover de nuevo (este particular va a cambiar en el siguiente capítulo).

|  |  |
| --- | --- |
| 10: Animaciones | 80 |

**Guardando nuestro movimiento**

Anteriormente, hemos sido capaces de actualizar estado inmediatamente después de que se desencadena la acción. Ahora tenemos que estirarlo con el tiempo, y deje de tomar de entrada mientras que estamos animando. Esta es la primera parte de eso, mostrando la forma en que bloqueamos la entrada, mientras que la animación está teniendo lugar.

* Se esta moviendo: **falso**,
* dirección: 'detenido',

3 movePacMan (dirección) {

4**Si**(**esta**.obtener('Se esta moviendo')|| **esta**.pathBlockedInDirection (dirección)) {

5*// hacer nada, sólo esperar a que pase*

6} **más** {

* **esta**.conjunto('dirección', Dirección)

8**esta**.conjunto('Se esta moviendo', **cierto**)

* **esta**.movementLoop ()

10}

11 },

El propósito de la sentencia if es muy simple: si ya estamos en movimiento, o si el camino está bloqueado ... no hacer nada! De lo contrario, iniciar el bucle de movimiento.

La forma en que se construyo la sentencia if puede ser un poco controversial ... es por lo general no es muy bueno para tener una primera cláusula en blanco en una sentencia if / else. Otra opción es la siguiente:

* movePacMan (dirección) {
* **dejar** inputBlocked= **esta**.obtener('Se esta moviendo')|| **esta**.pathBlockedInDirection (directi\3 en)
* **Si**(!inputBlocked) {
* **esta**.conjunto('dirección', Dirección)

6**esta**.conjunto('Se esta moviendo', **cierto**)

* **esta**.movementLoop ()
* }

9 },

Me gusta la primera versión porque es claro que Se esta moviendoconduce a un callejón sin salida, mientras que con la segunda versión que tenemos que analizar un poco más. Sin embargo, como he dicho anteriormente, por lo general es una mala forma de tener una primera cláusula en blanco en if / else. Estas son sólo algunas de las ventajas y desventajas que tendrá que hacer al escribir código.

También ayuda a que yo sepa lo que el código con el tiempo se verá así, y sé que con el tiempo hacer algo en la primera rama de la sentencia if / else.

Así que si usted no está en movimiento, y el camino no está bloqueado, hemos creado la dirección, establecemos Se esta moviendo a cierto, entonces corre movementLoop.

|  |  |
| --- | --- |
| 10: Animaciones | 81 |

**El Movimiento de bucle**

Esto es lo que el bucle de movimiento se ve así:

* frameCycle: 1,
* framesPerMovement: 30,

3 movementLoop () {

4**Si**(**esta**.obtener('FrameCycle')== **esta**.obtener('FramesPerMovement')) {

5**dejar** dirección= **esta**.obtener('dirección')

6**esta**.conjunto('X', **esta**.nextCoordinate ('X', Dirección));

7 **esta**.conjunto('Y', **esta**.nextCoordinate ('Y', Dirección)); 8

9**esta**.conjunto('Se esta moviendo', **falso**);

1. **esta**.conjunto('FrameCycle',1);
2. **esta**.processAnyPellets ();
3. } **más** {
4. **esta**.incrementProperty ('FrameCycle');
5. Ember.run.later (**esta**, **esta**.movementLoop, 1000-1060);
6. }

17

1. **esta**.pantalla clara();
2. **esta**.drawGrid ();
3. **esta**.drawPac ();
4. },

Hay un poco que hacer aquí, así que vamos a analizarlo por partes.

En primer lugar, hemos creado frameCycle y framesPerMovement. framesPerMovement está destinado a ser una constante, y frameCycletiene la intención de cambiar. En concreto, vamos a incrementarframeCycleen 1 cada vez que vamos a través del bucle. Aquí está el bucle simplificada, en sustitución de un montón de cosas con los comentarios y dejando sólo el esqueleto del bucle.

* movementLoop () {
* **Si**(**esta**.obtener('FrameCycle')== **esta**.obtener('FramesPerMovement')) {

3*// ... dirección de reinicio*

4**esta**.conjunto('Se esta moviendo', **falso**);

* **esta**.conjunto('FrameCycle',1);

6*pastillas //...process*

7} **más** {

* **esta**.incrementProperty ('FrameCycle');
* Ember.run.later (**esta**, **esta**.movementLoop, 1000-1060);

|  |  |
| --- | --- |
| 10: Animaciones | 82 |

1. }
2. *cosas //...drawing*
3. },

Verá que el núcleo es una sentencia if / else. SiframeCycle es menos que framesPerMovement, A continuación, se incrementará por 1, vuelva a ejecutar todo 33 milisegundos más tarde (1000-1060 implica 60 cuadros por segundo). Una vezframeCycle es igual a framesPerMovement, Reiniciamos todo- entorno Se esta moviendo en false, el establecimiento frameCycle a 1, pasar a la siguiente X y y coordina y comer el precipitado con processAnyPellets. También detener el bucle, ya que no lo llamamos de nuevo. Por supuesto, después de cada carrera a través del lazo, que redibujar todo.

**Ember.run.later**

Ember.run.later es una construcción que no hemos visto todavía. Hace casi exactamente lo que dice en el estaño que dice “Ember, ejecute esta tarde”. Los detalles son que el primer argumento es el alcance para ejecutar la función en, el segundo argumento es la función a ejecutar, y el tercer argumento es el número de milisegundos más tarde para ejecutarlo. Darse cuenta dethis.movementLoop no tiene () al final- que es porque no lo está funcionando en este momento, sólo poner la propia función como argumento por lo que se puede ejecutar más tarde.

Javascripters de la vieja escuela pueden notar que Ember.run.later llena un papel similar como setTimeout. Esto es verdad. Con Ember, que usted nunca tendrá que utilizar setTimeout Nunca más.

**Dibujo con FrameCycles**

Así que nuestro código es un bucle correctamente, tomando su tiempo- se puede ver que por el retraso entre la pulsación de un botón y ver el PAC movi- pero no es la animación todavía. Para que esto suceda, tendremos que hacer nuestradrawCircle funcionar tanto de direccionalidad y frameCycle.

Vamos a empezar por el paso en una dirección a la drawCircle función:

* drawPac () {
* **dejar** X= **esta**.obtener('X');

3**dejar** y= **esta**.obtener('Y');

4**dejar** radiusDivisor= 2;

5**esta**.drawCircle (x, y, radiusDivisor, **esta**.obtener('dirección'));

6 },

7

8 drawPellet (x, y) {

9**dejar** radiusDivisor= 6;

10 **esta**.drawCircle (x, y, radiusDivisor,'detenido');

|  |  |
| --- | --- |
| 10: Animaciones | 83 |

1. },
2. drawCircle (x, y, radiusDivisor, dirección) { */\*...\*/*}

los drawPellet función siempre pasará en detenido, Ya que las pastillas no moverse. losdrawPac función, por otro lado, pasará en la dirección de que el PAC se está moviendo actualmente.

Luego, dentro de la drawCircle función, cambiamos la forma pixelX y pixelY se calculan:

* drawCircle (x, y, radiusDivisor, dirección) {

2**dejar** CTX= **esta**.obtener('CTX')

3**dejar** squareSize= **esta**.obtener('SquareSize');

4

5**dejar** pixelX=(X+ 1/2 + **esta**.offsetFor ('X', Dirección))\*squareSize;

6 **dejar** pixelY = (Y + 1/2 + **esta**.offsetFor ('Y', Dirección)) \* squareSize; 7

8ctx.fillStyle = '# 000';

9ctx.beginPath ();

1. ctx.arc (pixelX, pixelY, squareSize/radiusDivisor, 0, Mates.PI \* 2, **falso**);
2. ctx.closePath ();
3. ctx.fill ();
4. },

Además de añadir 1/2 a X y y, También estamos añadiendo offsetFor. Eso es hasta qué punto el círculo está desplazado del centro, que se calcula using- adivinaron la ITframeCycle.

* offsetFor (coordenadas, dirección) {
* **dejar** frameRatio= **esta**.obtener('FrameCycle')/ **esta**.obtener('FramesPerMovement');

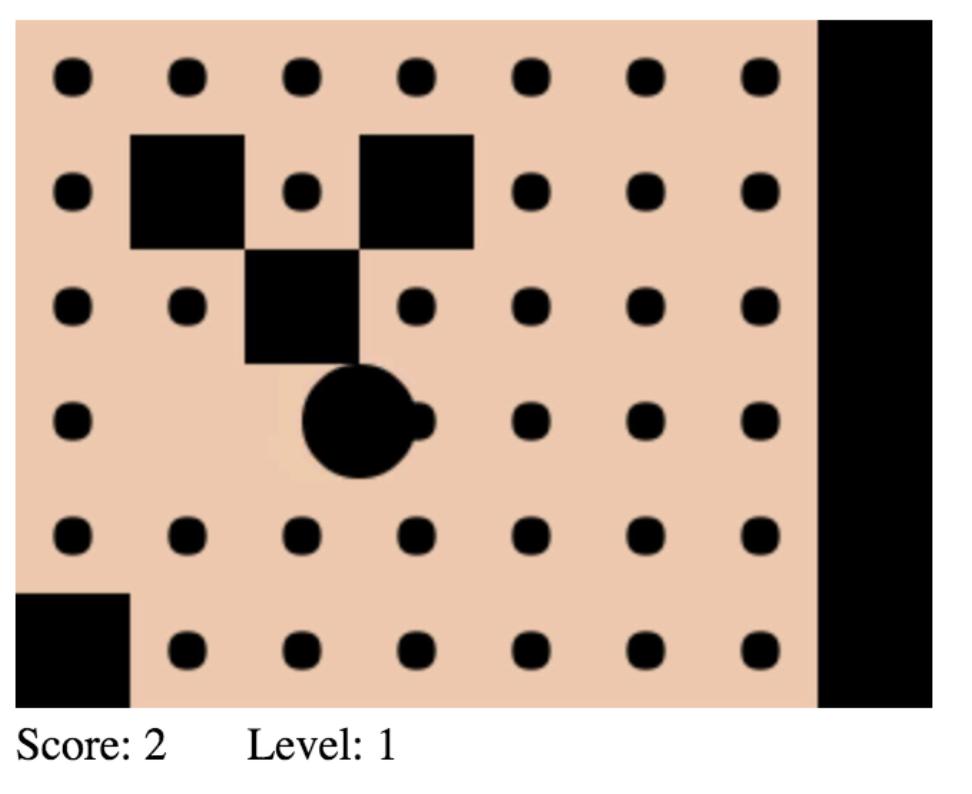
3 **devolver este**.obtener(`direcciones.**ps**dirección**}**.**ps**coordinar**}**`) \* frameRatio; 4 },

Así se obtiene la frameRatio, Que es básicamente el porcentaje del movimiento que se ha completado. Siempre será entre 0 y 1.

A continuación, se multiplica el frameRatio por el valor que obtenemos de la excavación en el direccionesUna vez más. Así que si estamos en el fotograma 10 (de 30), y nuestrodirecciónes 'correcto', entonces estaremos 1/3 del camino entre la plaza donde empezamos y el cuadrado a la derecha de la misma. Incrementando elframeCycle, Nos estamos moviendo el PAC frenar a la derecha, hacia su casilla de destino.

Y ahora tenemos un círculo de animación!

|  |  |
| --- | --- |
| 10: Animaciones | 84 |



**Resumen**

Vaya, así que fue el capítulo más largo, con mucho, y les pidió que pensar un poco bastante intenso. Felicitaciones por lo que a través!

En este capítulo se introduce la interpolación de cadenas, Ember.run.laterY el concepto de un bucle de animación. Además, le preguntamos a razonar profundamente sobre los hashes y hacer algunas refactors bastante complejas.

Todavía hay un montón de desafíos por delante, pero has demostrado que puede manejarlos.

**ninguna detención**

La adaptación a los nuevos controles se tomó un tiempo, pero fueron capaces de manejar bastante bien después de una hora. Matteo mantuvo dándole nuevos retos, cada una ligeramente más compleja que la anterior, y por tres horas en que era como si el PAC siempre se había movido de esta manera. Cuando finalmente salga de la PAC, los músculos se contraen para anticipar golpes y sacudidas que ya no están próximos.

La cena se extraño- una tensión tranquila se ha asentado sobre el grupo. El ambiente jovial vez que había excluido, y se sentía opresión en el momento, pero esto es aún peor. Hay rumores, a continuación, una, demasiado breve risa demasiado alta. El sonido de raspado de metal contra plástico.

Jerome se sienta a través de usted. “¿Cómo está la formación?”

“Era exactamente como usted ha dicho. Comenzamos el siguiente nivel de entrenamiento “.

“Bueno”, dice. Él parece que está a punto de decir algo más, pero entonces sólo asiente. “Bueno.” “¿Por qué todo el mundo está tan tranquilo?”

Mira a su alrededor, se inclina más cerca, susurra aún más silencioso. “Los fantasmas desaparecieron de Asia”. “Por lo tanto la resistencia realmente se ha ido.”

El asiente. “Yo no podría decir hoy en el almuerzo, pero ahora parece que todo el mundo sabe. Ningún uso que mantener aún más en la oscuridad “.

“Pero no sólo decir la resistencia se ha ido ... usted dijo que los fantasmas desaparecieron también.”

"Sí. Nuestro equipo de rastreo se había recogiéndolos en masa, pero ahora hay sólo unos pocos vagando por los túneles de pellets.”

“Mirando a los rezagados.”

"Exactamente. Nosotros no conocemos sus intenciones para antes tal vez era recursos, tal vez colonization- pero después de la conquista de Asia que acaba de recoger ya la izquierda. Matar y correr “.

“¿Y dónde están ahora?”

“Esa es la cosa ... no sabemos. Los que dejaron de Asia, que no pueden recogerlos más. Nadie puede."

Ambos en silencio termine su comida, sus pensamientos llenos de Armageddon. Eran los fantasmas de los ejércitos de Gog y Magog? Seguramente. Pero, de nuevo, Jesús no venía para dirigir sus ejércitos. Ambas religiones de acuerdo en que Él lo haría. Puedes adivinar que puede encontrar cuál es realmente cierto después de la batalla final, cuando vea si Toma a todos al paraíso o mata a todos los cerdos y rompe todos los cruces. Ya sea que te mate.

Estás dio un golpecito en el hombro, y dice que le han llamado para el entrenamiento adicional. Usted encontrará que no le importa.

85

|  |  |
| --- | --- |
| ninguna detención | 86 |

Matteo asiente en el PAC. Es sutilmente diferente a la que utilizó la última vez, y se encuentra con que no se detiene a intervalos regulares.

“Eficiencia”, dice cuando se le pregunte.

La formación continúa.

**11: El ciclo del juego**

En el último capítulo se alisó forma en que el PAC se mueve ... pero todavía paradas cada cuadrado! Eso no es óptimamente eficiente! En este capítulo vamos a hacer que lo que sigue rodando sin pulsar una dirección cada casilla. A continuación, vamos a introducir el sistema de 'intención' para que sea manejable.

**Rollin Rollin Rollin**

La razón de que el PAC se detiene la actualidad se debe a que el bucle se detiene ... una vez que llegue a la parte del código en el que mover las coordenadas del PAC y recoger las pastillas, que sólo se detiene y espera a instrucciones del reproductor. Podemos guardar cosas que están pasando por la introducción de un soloEmber.run.later en el otro camino dentro del bloque if / else.

movePacMan requiere un argumento para la dirección. En Ember.run.later, podemos proveer argumentos de la función pasada después de que pasan en la función, pero antes de que pase en el número de milisegundos de espera.

* movementLoop () {
* **Si**(**esta**.obtener('FrameCycle')== **esta**.obtener('FramesPerMovement')) {

3**dejar** dirección= **esta**.obtener('dirección')

4**esta**.conjunto('X', **esta**.nextCoordinate ('X', Dirección));

5 **esta**.conjunto('Y', **esta**.nextCoordinate ('Y', Dirección)); 6

* **esta**.conjunto('FrameCycle',1);

8**esta**.conjunto('Se esta moviendo', **falso**)

9

1. **esta**.processAnyPellets ();
2. Ember.run.later (**esta**, **esta**.movePacMan, dirección, 1000-1060)
3. } **más** {
4. **esta**.incrementProperty ('FrameCycle');
5. Ember.run.later (**esta**, **esta**.movementLoop, 1000-1060);
6. }

17

1. **esta**.pantalla clara();
2. **esta**.drawGrid ();
3. **esta**.drawPac ();
4. },

87

|  |  |
| --- | --- |
| 11: El ciclo del juego | 88 |

movePacMan, Si usted recuerda, llamadas movementLoop *Si* Se esta moviendo es falsa y el siguiente paso en la corriente dirección no esté bloqueada. Por lo tanto, siempre y cuando se cumplan las dos condiciones, el bucle sigue adelante.

El efecto neto de esto es que la PAC está fuera de control, manteniendo en ir como si fuera en el hielo. Esto podría hacer que para algunos puzzles interesantes, pero no es lo que queremos.

Pero antes de que se solucione este, vamos a hacer que sea aún más fuera de control:

* didInsertElement () {
* **esta**.movementLoop ();3 },

4 dirección: 'abajo',

Esto inicia el bucle automáticamente en lugar de esperar para la entrada, y se inicia en la dirección 'abajo'.

**El sistema de Intención**

La manera más fácil de dar al jugador el control es a responder de forma inmediata a sus demandas ... pero dentro de nuestro sistema de red, que sería un desastre. En su lugar vamos a implementar el sistema de 'intención', que almacena la siguiente dirección deseada del usuario (su intención) y luego actúa sobre que cuando se les da la oportunidad (cuando el PAC alcanza plenamente el siguiente cuadrado).

Nos ocuparemos de establecer la intención primera:

* intención: 'abajo',
* atajos de teclado: {
* arriba() { **esta**.conjunto('intención', 'arriba');},
* abajo() { **esta**.conjunto('intención', 'abajo');},

5izquierda() { **esta**.conjunto('intención', 'izquierda');},

6derecho() { **esta**.conjunto('intención', 'derecho');},

7 },

Así se inicia la intención 'abajo', y luego hemos cambiado las teclas de flecha para que cambien la intención en lugar de llamar directamente movePacMan.

Ahora vamos a renovar por completo movePacMan, Incluso el cambio de nombre changePacDirection para reflejar mejor sus nuevas responsabilidades.

|  |  |
| --- | --- |
| 11: El ciclo del juego | 89 |

* changePacDirection () {
* **dejar** intención= **esta**.obtener('intención')
* **Si**(**esta**.pathBlockedInDirection (intención)) {

4**esta**.conjunto('dirección','detenido');

5} **más** {

6**esta**.conjunto('dirección', La intención);

* }

8},

Encima de la tapa se dará cuenta de que hemos de cambiar el nombre movePacMan a changePacDirection. Eso es para reflejar el hecho de que es ahoramovementLoop haciendo el movimiento.

Después verá que estamos agarrando el intencióndel ámbito de aplicación de los componentes. Eso esintenciónque está siendo ajustado cuando llegamos a una tecla de flecha. La clave más recientemente alcanzó es lo que se usa. Así, comprobamos que laintención no nos va a ejecutar contra una pared, y luego cambiamos la dirección al intención. Si se ejecutará nosotros contra una pared, cambiamos dirección que se detuvo.

Por último, al final, hemos eliminado la llamada a movementLoop. Verá cuando lleguemos amovementLoop ¿por eso es posible.

Usted puede notar que hemos eliminado Se esta moviendo por completo, y la dirección 'parado' se hace cargo de muchos de los trabajos Se esta moviendo utilizado para manejar. Se esta moviendoera simplemente una solución intermedia introducida con el fin de mantener la curva de aprendizaje manejable al mismo tiempo hemos construido un sistema robusto. Espero que funcionó!

Ahora vamos a hacer nuestros cambios a movementLoop:

* movementLoop () {
* **Si**(**esta**.obtener('FrameCycle')== **esta**.obtener('FramesPerMovement')) {

3**dejar** dirección= **esta**.obtener('dirección')

4**esta**.conjunto('X', **esta**.nextCoordinate ('X', Dirección));

5 **esta**.conjunto('Y', **esta**.nextCoordinate ('Y', Dirección)); 6

* **esta**.conjunto('FrameCycle',1);
* **esta**.processAnyPellets ();
* **esta**.changePacDirection ();

1. } **else if**(**esta**.obtener('dirección') == 'detenido') {
2. **esta**.changePacDirection ();
3. } **más** {
4. **esta**.incrementProperty ('FrameCycle');
5. }

15

1. **esta**.pantalla clara();

|  |  |
| --- | --- |
| 11: El ciclo del juego | 90 |

1. **esta**.drawGrid ();
2. **esta**.drawPac ();
3. Ember.run.later (**esta**, **esta**.movementLoop, 1000-1060);
4. },

El primer cambio es que hemos eliminado la referencia a Se esta moviendo.

La segunda es que hemos añadido una nueva rama de la declaración if / else. se introduce este bloque cuando eldirecciónestá parado'. Esta lógica está tomando el lugar de laSe esta moviendo comprobar que solía ser en movePacMan. La gran diferencia es que ahora el bucle sigue rodando, dibujo cosas y esperando la la intención cambiar.

La última es que hemos pasado la llamada recursiva a movementLoophasta el final de la función. Eso significa que se llama a cada tiempo- no hay salir de este bucle.

El efecto de estos cambios es que cuando se pulsa un botón, se registra su intención, y luego el PAC comienza a moverse en esa dirección cuando se alcanza el siguiente cuadrado.

Vale la pena señalar que esto es muy diferente que el “bucle de ejecución” en la mayoría de los programas de Ember. Esto se debe a la programación de un juego es muy diferente en algunos aspectos que la programación de una aplicación web típica. Así que, aunque los dos son los bucles de ejecución, asegúrese de no mezclar el bucle de juego de correr y el bucle de ejecución Ember (aparte deEmber.run.later, No vamos a encontrar directamente el Ember bucle de ejecución en este libro).

**Resumen**

En este capítulo hemos completado nuestro bucle de juego, entonces se une que el sistema de la intención de proporcionar un sistema de control suave y avanzada.

Hemos logrado mucho en estos últimos 11 capítulos, pero cosas ha conseguido un poco difícil de manejar. NuestraPac-man.js archivo es más de 200 líneas! En los próximos 3 capítulos vamos a combinar las clases de Ember y mixins con sistema de módulos moderna de Javascript con el fin de separar parte de la lógica y hacer que el código sea más fácil de entender.

**objetivar**

Al llegar de nuevo colapsar en su litera. Otros tres horas de intenso entrenamiento han llevado a adelgazar, y te deja dormir casi de inmediato.

Pero no dura. 2 de la mañana se despierta con un sobresalto, media recordando un sueño con llamas. Una media hora más tarde se da cuenta de que el sueño no vendrá de nuevo, al menos no pronto, así que comience vagando por los pasillos.

Son diferentes por la noche. Las luces en las paredes se atenúan, una décima parte de la iluminación que por lo general son. Los seres humanos pueden vivir bajo tierra, pero las realidades de sus ritmos no habían cambiado. Esa fue la bendición de la tradición, la sabiduría del Islam. Al no reconocer el valor de la tradición había sido la caída de los infieles.

Pero sus pensamientos tienden a centrarse más en su propio downfall- el que ya había comenzado, que ahora se estaba extendiendo más allá de Asia. Se puede caminar y contemplar, trazando la mano a lo largo de la pared, hasta que escuche voces.

Al principio están en calma, un eco en la distancia. Se empieza a reconocer las voces antes de reconocer palabras, y se puede oír Jerome y Terrance, junto con una tercera voz que no reconoce, y una cuarta que viene de una radio.

“Todavía estamos aquí”, dice la voz en la radio. “No hay muchos de nosotros izquierda mil, tal vez, en Muharrad, y probablemente un poco más en algunas otras ciudades.”

“Quiero saber más sobre el final”, dice la tercera voz, desconocida. "¿Qué pasó exactamente?

“Ellos simplemente dejaron .... Sabían que estábamos aquí, tenían que saber que estuvimos aquí, pero justo arriba ya la izquierda.”

“Nuestros sensores todavía están recogiendo unos fantasmas aquí y allá.”

“Bueno, sí, hay algunos, pero no son vicioso como estaban antes. No más masacre ya que se retiraron. Nos están persiguiendo alrededor de metro, pero sólo cuando vamos detrás de los pellets.”

“Así que los civiles están a salvo?”

“Parecería que sí, señor. Pero no ha sido durante mucho tiempo. Pueden empezar de nuevo “.

“Eso es un alivio, por lo que va. Vamos a llamar de nuevo a las once de cien tienen los informes estándar listo para entonces. El Califato que gracias por su servicio “.

“Gracias, general.”

El zumbido de la radio se corta.

“Jerome, Terrance”, dice el general, “¿Cómo está su proyecto ... llegando a lo largo.”

“Él está progresando más rápido de lo esperado”, dice Terrance. “Incluso con nuestras expectativas ajustadas.”

91

|  |  |
| --- | --- |
| objetivar | 92 |

“Te lo dije”, dice Jerome.

“Yo tenía que divulgar algo de información clasificada. Sentí que era la mejor estrategia de mitigación en el momento.”‘Entendido’, dice el general. “Jerome, ¿qué hay en su final?”

“Nada clasificado que no era ya en la fábrica de rumores. Y no creo que sospecha.”‘Excelente, pero me refería en su salud mental.’

"Sí señor. Ha tenido una agenda muy apretada, y algunos golpes y contusiones, pero creo que está empujando a través. Él no es frágil “.

“Ha tenido algunos temblores PAC,” interpone Terrance.

Sintoniza a cabo como se llega a la conclusión: Jerome no es su amigo. Tal vez era, tal vez todavía lo es, pero sobre todo ahora que están a su sujeto de prueba, su muestra observada. Un acto.

“Su progresión mental también va bien. Él ha recogido una gran cantidad de nuevas formas de pensar, como hemos visto en los informes, pero aún no ha realmente refactorizado su cerebro para funcionar como la de un operador PAC “.

"Arregla eso. Suficiente para él para sobrevivir, por lo menos “.

"Sí señor."

Eventualmente, la conversación gira en torno a otros temas, menos interesantes. A medida que sigilosamente a su habitación, decide no dejar salir lo mucho que sabes. Si están ocultando algo, lo que puede.

Dormir era, de alguna manera, no más fácil.

**12: separación de las preocupaciones**

Nos dimos cuenta al final del capítulo anterior que el archivo componentes / PAC-man.jsera más de 200 líneas de largo. Eso es muy largo, y peor aún, se estaba haciendo difícil seguir la pista de dónde habíamos puesto varias funciones. Tal vez usted se pegóoffsetFor entre atajos de teclado y movementLoopen lugar de con los métodos de dibujo en las que tiene el sentido más lógico. O tal vez lo tienes todo en un orden lógico, pero hay tantas cosas que es difícil de recordar lo que era ese orden lógico.

En los dos capítulos siguientes vamos a dividir todo este código en varias secciones lógicas basadas en diferentes archivos y los que se accede de diferentes maneras. Esto añade un poco de cabeza, pero en última instancia hace nuestro trabajo más fácil- especialmente cuando más adelante queremos aumentar la complejidad del juego (por ejemplo, cuando se añade fantasmas).

En este capítulo se hará una revisión de los conceptos que necesitaremos saber con el fin de realizar esta separación, y el siguiente capítulo será la separación real.

**La definición de nuevos archivos**

Vamos a empezar por la definición de un Pac como un objeto separado Ember. El siguiente código es suficiente para empezar: “'js //models/pac.js importar Ember de 'brasa';

por defecto de exportación Ember.Object.extend ({}) “'

Esto es bastante similar a lo que teníamos cuando nos creamos nuestro componente:

* *//components/pac-man.js*
* **importar** Ember desde'ascua';

3

4 **por defecto de exportación** Ember.Component.extend ({})

Eso no es casualidad! Hay dos niveles de similitud aquí.

De ahora en adelante, ejemplos de código tendrán un nombre de archivo adjunto. Anteriormente, se supone que en los componentes / PAC-man.js menos que se especifique lo contrario.

**El retorno de los módulos ES2015**

El primer nivel de similitud es la forma básica de un archivo dentro brasa-cli. Importar cosas, entonces se exporta cosas- y por muchas cosas, que la exportación se realiza a través extender un objeto.

Antes de llegar al segundo nivel de similitud, vamos a seguir adelante y obtener el objeto Pac dentro de nuestro componente pac-man. Lo hacemos a través de la palabra clave Thata importación:

93

|  |  |
| --- | --- |
| 12: separación de las preocupaciones | 94 |

* *//components/pac-man.js*
* **importar** Ember desde'ascua';
* **importar** pac a partir'../models/pac';

El Pac que estamos importando es la exportación por defecto de ../models/pac. La ruta del archivo tiene un '..' en ella porque estamos utilizando las rutas de archivos relativos, y '..' significa 'ir a una carpeta'. También se podría utilizar una ruta absoluta del archivo:appName / modelos / pac.

Así, el resultado neto de utilizar módulos ES2015 (importación / exportación) es que podemos compartir código entre archivos.

Esa línea de importación es lo mismo que decir:

* **dejar** Pac=Ascua.Objeto.ampliar({})

Al llenar el objeto Pac con la funcionalidad (por no hablar de utilizarlo en varios contextos diferentes) y luego tener en un archivo separado se vuelve cada vez más útil.

**Sistemas con clase**

La segunda similitud entre nuestra declaración de objetos y nuestra declaración de componentes es que ambos están extendidos a través de un hash. En este momento la extensión de hash Pac está vacía, pero se llena de valores (en su mayoría funciones) unidos a distintas teclas, al igual que el hash de extensión de componentes es en este momento.

Esta similitud de la forma es porque Ember.Component en realidad es sólo una extensión de Ember.Object.

Ember.Component Se define como Ember.View.extend ({...}), Que se define como Ember.CoreView.extend ({...}), que se define por Ember.Object.extend ({...}). Por lo que un componente Ember es sólo un objeto Ember con un montón de material añadido a la misma.

En nuestra aplicación hasta el momento, las únicas partes de ese material añadido que hemos aprovechado hasta el momento son: 1) el hecho de que podemos pegar un componente en un manillar, y 2) que tenemos una plantilla adjunta con acceso a todas las variables dentro del alcance. Todo lo demás que hemos aprendido sobre el componente se aplica al objeto.

[Haga clic aquí para ver un video en el sistema de clases de Ember y la herencia chain¹¹](https://www.emberscreencasts.com/posts/40-class-inheritance)

Si usted nunca ha usado antes, pero Ember Algo de esto se busca familiar, es probablemente porque Ember ha reconstruido a propósito algo muy similar a la del sistema- objeto Ruby y, al hacerlo, han creado un sistema que es mucho más familiar para la mayoría (incluyendo los desarrolladores de otros lenguajes orientados a objetos como Java o Python) que prototipo de sistema por defecto de Javascript.

¹¹<https://www.emberscreencasts.com/posts/40-class-inheritance>

|  |  |
| --- | --- |
| 12: separación de las preocupaciones | 95 |

**mixins**

Hay algunas cosas que necesitaremos acceso a tanto en el componente de Pac-Man y el objeto pac. Para aquellos, usaremos un Mixin. . Mixins trabajo mediante la fusión de su hash con la del objeto dónde se mezclan en Por ejemplo, aquí hay un mixin se mezcle a tres objetos diferentes Ember:

* **dejar** PelletFiend=Ember.Mixin.extend ({

2lovesPellets: **cierto**

3 })

4 **dejar** PAC1 = Ascua.Objeto.extend (PelletFiend, {});

5 **dejar** PAC2 = Ascua.Objeto.extend (PelletFiend, {});

6 **dejar** SickPac = Ascua.Objeto.extend (PelletFiend, {

7lovesPellets: **falso**

8 })

Eso es exactamente el mismo que define a todos ellos de la siguiente manera:

* **dejar** PAC1=Ascua.Objeto.ampliar({

2lovesPellets: **cierto**

3 });

4 **dejar** PAC2 = Ascua.Objeto.ampliar({

5lovesPellets: **cierto**

6 });

7 **dejar** SickPac = Ascua.Objeto.ampliar({

8lovesPellets: **falso**

9 })

Como puedes ver, PAC1 y PAC2 asumir la lovesPellets: true valor de la PelletFiend mixin, mientras SickPac ya tiene un valor para el lovesPelletspropiedad, que tiene prioridad sobre el valor dado en la Mixin. Por supuesto, ya que nuestro ejemplo MixIn sólo tiene una propiedad definida, que es un poco tonto para que se mezcle a un objeto que ya se ha asignado un valor a la propiedad ... pero la gran mayoría de Mixins tener varias propiedades, algunas de las cuales no debe ser anulados, y algunos de los cuales específicamente marcados como valores por defecto cambiables como una forma de cambiar cómo se utiliza el Mixin.

De todas formas, en nuestro caso sólo va a utilizar el Mixin como un lugar para almacenar las propiedades y funciones que son necesarias tanto en el modelo Pac y el componente pac-man. Aquí es cómo definimos nuestra mixin desnuda:

|  |  |
| --- | --- |
| 12: separación de las preocupaciones | 96 |

* *//mixins/shared-stuff.js*

2 **importar** Ember desde 'ascua';

3

4 **por defecto de exportación** Ember.Mixin.extend ({})

Y así es como se importa en el componente pac-man:

* **importar** Ember desde'ascua';
* **importar** KeyboardShortcuts de'Brasa entre el teclado y atajos / mixins / componente';

3 **importar** SharedStuff de '../mixins/shared-stuff';

4 **importar** pac a partir '../models/pac';

5

6 **por defecto de exportación** Ember.Component.extend (KeyboardShortcuts, SharedStuff, {...})

Si usted ya es un ingeniero de software profesional, que probablemente murió un poco por dentro en el nombre de Mixin SharedStuff, Pero les aseguro que el nombre es temporal. Después de la separación es completa, y como añadimos una mayor complejidad al juego, vamos a crear una historia para compartir códigos con más matices (y mejor llamado-).

Nombrando a un lado, lo importante que quería mostrar era que nos mezclando nuestra Mixin en el componente. Se necesita la misma forma que cuando se mezclan en KeyboardShortcuts, pero ahora en lugar de mezclar en el código de otra persona de un complemento que estamos mezclando en nuestro propio código desde dentro del proyecto.

**Las clases e instancias**

Hasta ahora, en este capítulo hemos cubierto cómo dividir el código y luego traerlo de vuelta juntos- dividirlo en objetos y Mixins, luego traerlo de vuelta junto con el sistema de módulos ES2015 (declaraciones de importación / exportación). Eso es casi lo suficiente para poder hacer frente a refactor del siguiente capítulo, pero hay un aspecto más del sistema de objetos que tendremos que hacer frente.

Digamos que nos movemos la variable Xdesde el componente pac-man al modelo Pac. Queremos hacer eso porqueXrepresenta el valor x de donde el Pac es, por lo que tiene sentido tener que dentro del objeto Pac. Sin embargo, eso nos causa un problema- necesitamos acceso aX! Tenemos que averiguar cómo el componente pac-man será capaz de acceder a la variableX en el Pac.

Hemos importado Pac utilizando el sistema de módulos, pero si tratar de acceder al valor de x utilizando Pac.get ( 'x')(O cualquier otro método directo que se pueda imaginar) no va a funcionar. Esto se debe a laPacque importamos es la definición de clase. No es un objeto Pac misma- sólo nos dice cómo crear objetos Pac.

Así es como podemos tener acceso a esa X valor:

|  |  |
| --- | --- |
| 12: separación de las preocupaciones | 97 |

* **importar** pac a partir'../models/pac';

2

3 **por defecto de exportación** Ember.Component.extend (KeyboardShortcuts, SharedStuff, {

4didInsertElement () {

5**dejar** pac=Pac.create ()

6pac.get ('X') *// 1*

7**esta**.conjunto('PAC', Pac)

* **esta**.movementLoop ();

9},

1. anotherFunction () {
2. **esta**.obtener('Pac.x')*// 1*
3. }
4. })

Esto es lo que hemos hecho:

1. Creado un objeto Pac, que es una instancia de la clase Pac.
2. Llamado obteneren ese caso, lo que nos dio el valor de cualquier propiedad en el objeto. En este caso,X (Y el valor devuelto es 1)
3. Poner el ejemplo en el ámbito de componentes por lo que podemos acceder desde cualquier lugar dentro del componente
4. Se accede desde otra parte del componente (y utilizado profunda llegando a alcanzar directamente la X valor)

Para aquellos que prefieren las analogías, imaginar que el Pacdefinición de clase importamos es un cortador de galletas. Cuando llamamoscrearcon la definición de clase, es como poner el cortador de galletas hacia abajo en masa de galletas laminado plano y crear una nueva cookie. La cookie es la instancia y el cortador de galletas es la definición de clase.

**Resumen**

En este capítulo hemos aprendido acerca de Object System de Ember, que tiene una sorprendente similitud con el Sistema de objetos de Ruby. Una especie de locura que un marco Web está recreando partes de otros idiomas, pero bueno, es Javascript!

En el siguiente capítulo vamos a utilizar nuestra recién descubierta conocimiento de objetos, Mixins, clases y módulos ES2015 con el fin de refactorizar nuestro código en algo mucho más manejable.

**lealtades**

Su objetivo es actuar de la misma, como si no hubiera oído nada anoche ... pero aún así, todavía no se puede dejar de ser más reticentes al hablar con Jerome. La sensación no es la misma.

Él no es el enemigo, se dice a sí mismo. Él no es el enemigo ... sólo un espía.

“¿Cómo fue la sesión de entrenamiento de anoche”, se pregunta.

En respuesta, usted Pala comida en la boca. No ayuda.

“Este es un momento complicado durante el entrenamiento. Puede que no se siente como mucho está cambiando, pero usted está absorbiendo una gran cantidad de nuevos patrones. Es necesario asegurarse de que usted le da tiempo para gel, tiene paciencia, incluso si es complicado “.

A reconocer las palabras de la noche anterior. Esta charla sido pre-ensayado. Los otros, los que le inspiraron mucho antes- también pre-ensayado. Todavía se mantiene la idea de que todo antes de este trabajo había sido genuina, pero mirando a Jerome hablando se ve tal como lo hizo en aquel entonces.

Usted se levanta y camina hacia la otra mesa. En el camino, se lanza hacia arriba.

Nunca lo hace a la otra mesa- se termina retirándose al baño en su lugar.

Jerome sigue, y se siente ganas de vomitar de nuevo.

“Fuck off”, dice usted.

Se realiza copias de seguridad. “Mala comida?”, Pregunta con nerviosismo. Usted está apoyado contra la pared de azulejos, pero él es el que copia de seguridad y nervioso.

“Sé lo que estás haciendo”, dice usted.

“Estoy ayudando a un amigo, eso es lo que estoy haciendo.”

"Mierda. Te oí hablando con ellos la noche anterior. Estás espiándome “.

“Soy yo o alguien más. Los convencí de que sería más eficaz, puesto que ya se sabía.”“Felicidades. Usted es eficaz. ¿Quieres que yo sea agradecido?”

“Te estoy protegiendo, Salam. ¿Quieres otra persona rebuscar en sus libros? ¿Quieres otra persona viendo sus labios cuando oras? Conseguir que en el vídeo, tratando de psicoanalizar?”

"¿Qué estás implicando?"

“Usted es un apóstata, Salam. Eres un infiel “.

“Estoy atento.”

98

|  |  |
| --- | --- |
| lealtades | 99 |

“Así soy yo, y lo que ha observado en el laboratorio, cuando eras más descuidados. No eres 'Ahl al-Kitab. Usted no consigue ser otra cosa que lo que nació “.

“No les importa. Observo las formas, no lo anuncian. Además, soy demasiado importante.”Usted dice que es de esperar. Con determinación.

“Tal vez ahora, pero ¿qué pasa cuando consiguen su arma? ¿Crees que te tratan como un dhimmi, cuando usted no nació dhimmi? ¿Que hay de tus padres? ¿Tus hermanas? No va a terminar bien, Salam “.

Su corazón se hunde. Usted sabe que tiene razón.

El año pasado, José © se había encontrado con un antiguo tomo de uno de los natives- algo Harris, que puso en duda la existencia de Dios. Se decía que era un recuerdo de su abuela. Pero aún así, no regresó al laboratorio. Su único mensaje de él desde entonces fue un video curiosamente redactado, una despedida, diciendo que estaba transfiriendo a un laboratorio secreto.

Uno se da cuenta, con creciente horror lentamente, que no había laboratorio secreto.

Es hora de compartimentar. Hacen sus ideas separate- totalmente privado para los extraños. Jerome se mira a los ojos y cabeza. “Voy a quemar la misma.”

**13: separación de las preocupaciones**

En este capítulo vamos a separar a cabo una Pac objeto de la pac-man componente.

Estamos haciendo esto por varias razones. La primera es que lapac-mancomponente, actualmente en 200 líneas, es sólo estamos demasiado larga. La segunda es que tener un objeto en el que almacenar todas las variables y métodos relacionados con el PAC-sólo tiene sentido. Vamos a realmente encantados de esta separación cuando añadimos en la complejidad de los fantasmas (y vidas extra).

**Separación: Objeto Pac**

El plan de juego es separar la Pac, Poner lo que tiene que ser compartida en el SharedStuff mixin, y luego editar varias funciones en el pac-man componente para que correctamente los valores de referencia como en el Pac ejemplo.

Vamos a empezar con el Pac.

* **importar** Ember desde'ascua';
* **importar** SharedStuff de'../mixins/shared-stuff';

3

4 **por defecto de exportación** Ascua.Objeto.extend (SharedStuff, {

5dirección: 'abajo',

* intención: 'abajo',

7

* X: 1,

9y: 2,

10

1. dibujar(){
2. **dejar** X= **esta**.obtener('X');
3. **dejar** y= **esta**.obtener('Y');
4. **dejar** radiusDivisor= 2;
5. **esta**.drawCircle (x, y, radiusDivisor, **esta**.obtener('dirección'));
6. },

17

1. cambia la direccion(){
2. **dejar** intención= **esta**.obtener("intención")
3. **Si**(**esta**.pathBlockedInDirection (intención)) {
4. **esta**.conjunto('dirección','detenido');
5. } **más** {

100

|  |  |
| --- | --- |
| 13: separación de las preocupaciones | 101 |

1. **esta**.conjunto('dirección', La intención);
2. }
3. },

26

1. pathBlockedInDirection (dirección) {
2. **dejar** cellTypeInDirection= **esta**.cellTypeInDirection (dirección);
3. **regreso** Ember.isEmpty (cellTypeInDirection)||cellTypeInDirection=== 1;
4. },

31

1. cellTypeInDirection (dirección) {
2. **dejar** nextX= **esta**.nextCoordinate ('X', Dirección);
3. **dejar** nextY= **esta**.nextCoordinate ('Y', Dirección);
4. **devolver este**.obtener(`Cuadrícula.**ps**nextY**}**.**ps**nextX**}**`);
5. },
6. nextCoordinate (coordenadas, dirección) {
7. **devolver este**.get (coordenadas)+ **esta**.obtener(`direcciones.**ps**dirección**}**.**ps**coordinar\
8. **}**`);
9. },
10. })

Este es nuestro Pacobjeto. Nótese que las variables que hemos traído a lo largo de:dirección, intención, Xy y. Esas son todas las variables que se refieren directamente al estado de la Pac (más que el estado del resto del mundo).

También hemos traído a lo largo de algunas funciones. Tomar nota de dos de los:dibujar y cambia la direccion. El astuto se dará cuenta de que solían ser llamadosdrawPac y changePacDirection- tener el nombre de la clase en el nombre de la función es un claro indicativo de que usted debe transferirlo hacia el objeto.

Los tres últimos fueron llevados funciones a lo largo ya que se necesitaban para cambia la direccion-

cambia la direccion llamadas pathBlockedInDirection, Que llama cellTypeInDirection, Que llama nextCoordinate. Si se hubieran quedado en el componente de pac-man que no sería capaz de acceder fácilmente desde dentro del Pac.

Los lectores que son de doble control mis explicaciones pueden haber notado una discrepancia en ese último párrafo- cambia la direccion necesariamente pathBlockedInDirection para estar dentro del Pac alcance, pero entonces ¿por qué es dibujar permitido llamar drawCircle, Que no hemos incluido en el Pac¿definición? Por lo demás, ¿qué pasa con eldirecciones hash y la cuadrícula, Ambos de los cuales se accede de cambia la direccion dependencias? La respuesta está en el ShareStuff mixin.

|  |  |
| --- | --- |
| 13: separación de las preocupaciones | 102 |

**Separación: SharedStuff Mixin**

los SharedStuff mixin es donde estamos poniendo todo lo que se necesita por tanto la pac-man componente y el Pac objeto.

Normalmente SharedStuffes un nombre genérico horrible para un mixin, pero en este caso se ajusta, porque lo único que mantiene estas cosas juntas temáticamente es que son cosas que se comparte tanto con el componente y el objeto. A medida que el intercambio se refinó, también lo hará el nombre.

Aquí encontraremos la cuadrícula propiedad, el direcciones propiedad, y el drawCircle función que nos 'perdido' en el Pac definición, así como las propiedades y funciones necesarias para apoyarlos.

* **importar** Ember desde'ascua';

2

3 **por defecto de exportación** Ember.Mixin.create ({

* frameCycle: 1,
* framesPerMovement: 30,

6

7*// 0 es un espacio en blanco*

8*// 1 es una pared*

9*// 2 es un pellet*

1. cuadrícula: [
2. [2, 2, 2, 2, 2, 2, 2, 1],
3. [2, 1, 2, 1, 2, 2, 2, 1],
4. [2, 2, 1, 2, 2, 2, 2, 1],
5. [2, 2, 2, 2, 2, 2, 2, 1],
6. [2, 2, 2, 2, 2, 2, 2, 1],
7. [1, 2, 2, 2, 2, 2, 2, 1],
8. ],
9. squareSize: 40,

19

1. CTX: Ember.computed (**función**() {
2. **dejar** lona=documento.getElementById ("myCanvas");
3. **dejar** CTX=canvas.getContext ("2D");
4. **regreso** CTX;
5. }),

25

1. drawCircle (x, y, radiusDivisor, dirección) {
2. **dejar** CTX= **esta**.obtener('CTX')
3. **dejar** squareSize= **esta**.obtener('SquareSize');

|  |  |
| --- | --- |
| 13: separación de las preocupaciones | 103 |

1. **dejar** pixelX=(X+ 1/2 + **esta**.offsetFor ('X', Dirección))\*squareSize;
2. **dejar** pixelY=(Y+ 1/2 + **esta**.offsetFor ('Y', Dirección))\*squareSize;
3. ctx.fillStyle = '# 000';
4. ctx.beginPath ();
5. ctx.arc (pixelX, pixelY, squareSize/radiusDivisor, 0, Mates.PI \* 2, **falso**);
6. ctx.closePath ();
7. ctx.fill ();
8. },
9. offsetFor (coordenadas, dirección) {
10. **dejar** frameRatio= **esta**.obtener('FrameCycle')/ **esta**.obtener('FramesPerMovement');
11. **devolver este**.obtener(`direcciones.**ps**dirección**}**.**ps**coordinar**}**`)\*frameRatio;
12. },

43

1. direcciones: {
2. 'arriba': {X: 0, y: -1},
3. 'abajo': {X: 0, y: 1},
4. 'izquierda': {X: -1, y: 0},
5. 'derecho': {X: 1, y: 0},
6. 'detenido': {X: 0, y: 0}
7. },
8. });

Aunque Pac Sólo llamadas directamente dos propiedades y una función, tenemos seis propiedades (uno es una propiedad computada) y dos funciones en el Mixin. Si estábamos diseñando una API que podríamos etiquetar las dos propiedades y la función que llamamos como público, y los otros como privado.

**Separación: Componente pac-man**

Ahora vamos a volcar la totalidad pac-mancomponente aquí. Así que lo siento, pero tiene que ser hecho. Hice mostrar un poco de misericordia mediante la compactación de las funciones que no se cambian con una....

Explicación a seguir.

|  |  |
| --- | --- |
| 13: separación de las preocupaciones | 104 |

* **importar** Ember desde'ascua';
* **importar** KeyboardShortcuts de'Brasa entre el teclado y atajos / mixins / componente';

3 **importar** SharedStuff de '../mixins/shared-stuff';

4 **importar** pac a partir '../models/pac';

5

6 **por defecto de exportación** Ember.Component.extend (KeyboardShortcuts, SharedStuff, {

7didInsertElement () {

8**esta**.conjunto('PAC', Pac.create ())

* **esta**.movementLoop ();

10},

11

12Puntuación: 0,

13levelNumber: 1,

14

15ScreenWidth: Ember.computed (**función**() {...}),

dieciséisScreenHeight: Ember.computed (**función**() {...}),

17screenPixelWidth: Ember.computed (**función**() {...}),

18screenPixelHeight: Ember.computed (**función**() {...}),

19

20drawWall (x, y) {...},

21DibujarRejilla () {...},

22drawPellet (x, y) {

23**dejar** radiusDivisor= 6;

24**esta**.drawCircle (x, y, radiusDivisor,'detenido');

25},

26

27pantalla clara(){...},

28

29movementLoop () {

30**Si**(**esta**.obtener('Pac.frameCycle')== **esta**.obtener('Pac.framesPerMovement')) {

31**dejar** dirección= **esta**.obtener('Pac.direction')

32**esta**.conjunto('Pac.x', **esta**.obtener('PAC') .nextCoordinate ('X', Dirección));

33 **esta**.conjunto('Pac.y', **esta**.obtener('PAC') .nextCoordinate ('Y', Dirección)); 34

1. **esta**.conjunto('Pac.frameCycle',1);
2. **esta**.processAnyPellets ();
3. **esta**.obtener('PAC').cambia la direccion();
4. } **else if**(**esta**.obtener('Pac.direction') == 'detenido') {
5. **esta**.obtener('PAC').cambia la direccion();
6. } **más** {

|  |  |
| --- | --- |
| 13: separación de las preocupaciones | 105 |

1. **esta**.incrementProperty ('Pac.frameCycle');
2. }

45

1. **esta**.pantalla clara();
2. **esta**.drawGrid ();
3. **esta**.obtener('PAC').dibujar();
4. Ember.run.later (**esta**, **esta**.movementLoop, 1000-1060);
5. },
6. processAnyPellets () {
7. **dejar** X= **esta**.obtener('Pac.x');
8. **dejar** y= **esta**.obtener('Pac.y');
9. **dejar** cuadrícula= **esta**.obtener('cuadrícula');
10. **Si**(GRID [y] [x]== 2) {
11. rejilla [y] [x] = 0;
12. **esta**.incrementProperty ('Puntuación')
13. **Si**(**esta**.Nivel completado()){
14. **esta**.incrementProperty ('LevelNumber')
15. **esta**.volver a empersar el nivel()
16. }
17. }
18. },

68

1. Nivel completado(){...},
2. volver a empersar el nivel(){
3. **esta**.conjunto('Pac.x',0);
4. **esta**.conjunto('Pac.y',0);
5. **dejar** cuadrícula= **esta**.obtener('cuadrícula');
6. grid.forEach ((fila, rowIndex)=>{
7. row.forEach ((célula, columnIndex)=>{
8. **Si**(celda== 0) {
9. rejilla [rowIndex] [columnIndex] = 2
10. }
11. })
12. })
13. },

84

|  |  |
| --- | --- |
| 13: separación de las preocupaciones | 106 |

1. atajos de teclado: {
2. arriba() { **esta**.conjunto('Pac.intent', 'arriba');},
3. abajo() { **esta**.conjunto('Pac.intent', 'abajo');},
4. izquierda() { **esta**.conjunto('Pac.intent', 'izquierda');},
5. derecho() { **esta**.conjunto('Pac.intent', 'derecho');},
6. },
7. });

Vamos a empezar desde la parte superior.

Primero en didInsertElement, Estamos creando una instancia de Pac y se establece a pac en el ámbito de aplicación de los componentes.

A continuación, en drawPellet estamos accediendo a la drawCircle función, que ha sido trasladado a la SharedStuff mixin. No es un cambio, pero sólo mostrar que todavía podemos acceder. La misma historia para el CTX propiedad computarizada que está en el drawWall, DibujarRejillay pantalla clara funciones (funciones definiciones reducen al mínimo para ahorrar espacio).

Vamos a saltar movementLoop por ahora e ir directamente a processAnyPellets. Aquí estamos para accederpac.x y pac.y- como puedes ver, pac-man se puede acceder fácilmente a las propiedades que se encuentran en el Pacobjetar cuando se necesita. Envolver a empersar el nivel y los diversos atajos de teclado métodos vemos que también podemos establecer las propiedades del objeto Pac.

Ahora para movementLoop. En primer lugar, podemos verthis.get ( 'pac'). nextCoordinate, this.get ( 'pac'). changeDirect y this.get ( 'PAC'). draw ()que podemos llamar a las funciones que se encuentran en la instancia pac. En segundo lugar,

Observe que la función está llamando pac mucho- parece como si la instancia pac se hace referencia a cada línea.

De hecho, si se cuenta, verá que la pacejemplo, se hace referencia a mucho más que cualquier otra línea. Al igual que la palabraPac en movePac y changePacDirection, Esto es una señal de que la función puede pertenecer a la Pac clase.

Sin embargo, hay más matices aquí- DibujarRejilla, pantalla claray processAnyPellets claramente no pertenecen a la Pac. ¿Asi que que hacemos?

**Separación: extracción parcial método**

La respuesta es separar parte del bucle y ponerlo en un método en el Pac.

Esto es lo que dejamos en el pac-man componente:

|  |  |
| --- | --- |
| 13: separación de las preocupaciones | 107 |

* lazo(){
* **esta**.obtener('PAC').movimiento();

3

* **esta**.processAnyPellets ();

5

* **esta**.pantalla clara();
* **esta**.drawGrid ();
* **esta**.obtener('PAC').dibujar();

9

1. Ember.run.later (**esta**, **esta**.lazo, 1000-1060);
2. },

Este código es mucho más claro!

Hay dos puestos de trabajo se completará aquí: procesamiento de pastillas y dibujar cosas. El resto se deja al bucle en elPac objeto.

La mayor parte de nuestra funcionalidad es en el recién creado movimiento función, pero me he ido por delante y creó otras dos funciones con nombre con el fin de aclarar lo que estamos haciendo en el bucle:

* *//models/pac.js*

2 movimiento(){

* **Si**(**esta**.animationCompleted ()) {
* **esta**.finalizeMove ();
* **esta**.cambia la direccion();
* } **else if**(**esta**.obtener('dirección') == 'detenido') {
* **esta**.cambia la direccion();

8} **más** {

* **esta**.incrementProperty ('FrameCycle');

10}

11 },

12

13 animationCompleted () {

14 **devolver este**.obtener('FrameCycle') == **esta**.obtener('FramesPerMovement'); 15 }, dieciséis

17 finalizeMove () {

18**dejar** dirección= **esta**.obtener('dirección');

19**esta**.conjunto('X', **esta**.nextCoordinate ('X', Dirección));

20 **esta**.conjunto('Y', **esta**.nextCoordinate ('Y', Dirección)); 21

22**esta**.conjunto('FrameCycle',1);

23 },

|  |  |
| --- | --- |
| 13: separación de las preocupaciones | 108 |

La suma de animationCompleted y finalizeMove hacerlo más claro cómo movimiento lleva a cabo la tarea de mover el Pac. Hay varias partes de nivel inferior a ese trabajo, pero han sido en su mayoría arrendados a otras funciones: decidir si la animación se ha completado todavía, finalizando el movimiento, el cambio de laPac'S dirección, y continuando la animación si en medio de un movimiento.

Esto es mucho más claro que nuestra versión antigua! También demuestra un matiz importante en nuestra prácticas- refactorización cuando se está separando un objeto, la función no es la unidad más pequeña de separación. Usted puede terminar separando una función deficiente de puesta en común, en varias funciones -across más de un objeto- que terminan siendo mucho más coherente y comprensible.

Si estás siguiendo junto con el repositorio GitHub, lamento informarle que ahora han divergido. Esto se debe a que tenía previamente una por separado y de forma independiente a ejecutarlazoen el PAC que no sólo era confuso, pero fue de bajo rendimiento (especialmente después de varios reinicios del juego). Si hay un error de los próximos capítulos, es probable que a partir del cambio. Por favor enviar esos fallos a mí en bilesjeffrey@gmail.com.

Sin embargo, hay un problema que surge de todo esto.

**Resyncing en reinicio**

El problema aflora al reiniciar el nivel, mientras que ir a la izquierda o hacia arriba. A veces, cuando se hace eso, el Pac irá fuera del área de juego un cuadrado.

Esto se hace debido a que el ciclo de animación y los bucles de pellets contando no se sincronizan para arriba. Anteriormente, pellet de conteo sucedió en entre la finalización de la jugada, decidir sobre la siguiente dirección. Ahora, el paso es decidir sobre laPacs movimientofunción, se decidirá en la siguiente dirección antes de que los gránulos pueden ser procesados ​​(y por tanto el nivel renovadas). Por lo que ya ha decidido a seguir adelante izquierda o hacia arriba cuando se reinicia el nivel (colocándolo en[0, 0]), Y se mantiene en movimiento hacia la izquierda o hacia arriba hasta que el ciclo se ha completado la animación.

Una forma de evitar esto es para mover el processAnyPellets llamar al Pac's movimiento función, pero eso significaría que lleva el processAnyPellets función y todas sus dependencias en el Pac. Un argumento podría hacerse de cualquier manera (es elPac que procesa los pellets), pero las dependencias (que dependen en gran medida el nivel y el estado actual de la red) realizar la transferencia inestable en este momento.

La forma en la que vamos a ir sobre ella es simplemente agregar algunas instrucciones adicionales a la volver a empersar el nivel método.

|  |  |
| --- | --- |
| 13: separación de las preocupaciones | 109 |

* volver a empersar el nivel(){
* **esta**.conjunto('Pac.x',0);

3**esta**.conjunto('Pac.y',0);

* **esta**.conjunto('Pac.frameCycle',0);
* **esta**.conjunto('Pac.direction','detenido')

6

7**dejar** cuadrícula= **esta**.obtener('cuadrícula');

8grid.forEach ((fila, rowIndex)=>{

9row.forEach ((célula, columnIndex)=>{

1. **Si**(celda== 0) {
2. rejilla [rowIndex] [columnIndex] = 2
3. }
4. })
5. })
6. },

Las instrucciones adicionales son para establecer pac.frameCycle a 0 y pac.directiona 'parado'. Que restablece esencialmente el bucle, lo que obligó a decidir sobre su dirección utilizando el nuevo estado del nivel.

**Resumen**

Eso fue mucho trabajo, pero espero que estará de acuerdo que el código más limpio, más comprensible hace que valga la pena. En el siguiente capítulo vamos a seguir refactorización del código sacando unaNivel clase, y luego finalmente vamos a empezar a ver los frutos de nuestro trabajo a medida que descubrimos una manera de crear múltiples niveles y cambiar dentro y fuera.

**Separación**

En realidad no se queme.

Eso sería desordenado. Incriminatorio. Simplemente no hay buena manera de encender discretamente un manuscrito en el fuego mientras que en un búnker subterráneo.

En su lugar, Ferry a la letrina y dejar que el agua penetre lentamente lejos sus huellas dactilares. Usted ras, para acelerar el proceso a lo largo de, y valorar después en un ácido. Alguien lo encontrará, pero para entonces las palabras se ha ido de las páginas, y será imposible de encontrar en su caso.

A medida que se aleja, se repite pasajes clave en su cabeza, asegurándose de que no se habían perdido realmente.

Aún así, cuando vuelvas a tu litera, llorar.

En el desayuno del día siguiente a evitar Jerome. Incluso con sus razones, se obtiene una sensación de malestar cuando lo veas.

Se puede caminar por delante de él, y caminar sin rumbo entre las mesas.

“Usted, chico.” Que a su vez y ver uno de los hombres fornidos PAC, sobre todo uno que tenía miedo que la primera vez que tienes aquí. No parecía tan intimidante ahora, ya que había amontonadas encima de un pedacito de su entrenamiento. Él también estaba sonriendo. "Toma asiento."

Otro hombre se ríe. “Alguien está tomando un paso para salir de su torre de marfil.”

“Pero sigue siendo tan grave”, dice otro. “No creo que desaparezca tan rápido.”

Usted se sienta lentamente, con cuidado, observando sus expresiones como lo hace. Sus ojos son alentadores, incluso si sus palabras son de burla.

“Tal vez él sabe algo acerca de los extraterrestres. Tal vez por eso no se nos ha unido “.

“¿Qué, como él sabe cuáles de nosotros va a morir, por lo que no quiere llegar a conocernos y obtener toda triste cuando sucede?”

“No, quiero decir ... mierda como grave. Tal vez en lugar de hacer el payaso con nosotros, está fuera de pensar en la estrategia y la salvación del mundo y la mierda “.

Él le da un aspecto importante, obviamente esperando que diga algo. Eres sin habla, tanto por la forma en que están hablando cuando estás ahí, haciendo chistes a su costa, y por lo que parece que han estado esperando para hablar con usted todo el tiempo. ¿Era realmente ustedes que había sido cerrada?

110

|  |  |
| --- | --- |
| Separación | 111 |

“Tal vez es un mudo”, dice otro.

Sí, definitivamente se había cerrado. Usted balbucear algo sobre los extranjeros, tratando de no revelar demasiado. “Los extraterrestres son ... sí, los extranjeros son un problema. Y yo he estado perdiendo mis amigos en el laboratorio “.

Ellos se ponen serias, de repente. “Extraño a mis mujeres y niños cada día que estoy aquí”, dice el hombre grande. “Lo único que me recibe a través de mis hijos está aquí.” Le hace un gesto con el resto de la mesa.

Otro hombre asiente con la cabeza. “Tengo una esposa y cuatro hijos en casa. La última vez que volvió, mi hija mayor me estaba contando la historia del cuerpo de PAC, preguntando cuando podría unirse a mí aquí “.

“No está casada como estos tipos”, dice uno de los hombres más jóvenes, “pero me dio la gente en la mezquita que echo de menos”.

“Oye, ¿sabes qué? Estamos Recopilación de pellets de hoy, pero unirse a nosotros en esta noche Isha'a. Un hombre no debe orar a solas.”Los otros están de acuerdo, un coro de aliento.

El desayuno se enciende, el ir y venir, y de vez en cuando participar. No son que usted está acostumbrado a estar alrededor- son crudo, en voz alta, y nada especial en términos de iQ, pero por primera vez en semana casi se puede olvidar todo lo que ha pasado.

Tu sonríes.

No dura.

Matteo es todo Business- claramente algo está pasando en el frente extranjero, pero no se lo dirá.

“Ese no es su preocupación. ¿Cuál es su preocupación es ser capaz de moverse entre los niveles, para adaptarse cuando se le da un nuevo diseño. ¿Estás listo?"

Asientes. Apretar los dientes.

Ningún ensayo os alcanzará más allá de lo que pueden soportar.

**14: Nivel de salida**

El último capítulo comenzó nuestra búsqueda para limpiar el componente pac-man separando secciones coherentes de código y haciéndolos sus propios objetos. Sacamos laPac objeto, así como la SharedStuff mixin.

En este capítulo se continúa esa búsqueda, sacando el Nivel oponerse, hacer algo más sentido de lo SharedStuff es, y luego usando nuestros objetos de nivel separadas recientemente para crear fácilmente intercambiables niveles alternos.

**El objeto de nivel**

Vamos a empezar por mostrar el objeto de nivel que nos hemos tirado a cabo, a continuación, explicar lo que se fue a las decisiones tomadas en lo que tirar y qué no para tirar, y luego mostrar qué cambios deben hacerse en otro lugar con el fin de mantener las cosas funcionando.

* *//models/level.js*
* **importar** Ember desde'ascua';

3

4 **por defecto de exportación** Ascua.Objeto.ampliar({

5*// 0 es un espacio en blanco*

6*// 1 es una pared*

7*// 2 es un pellet*

8cuadrícula: [

9[2, 2, 2, 2, 2, 2, 2, 1],

1. [2, 1, 2, 1, 2, 2, 2, 1],
2. [2, 2, 1, 2, 2, 2, 2, 1],
3. [2, 2, 2, 2, 2, 2, 2, 1],
4. [2, 2, 2, 2, 2, 2, 2, 1],
5. [1, 2, 2, 2, 2, 2, 2, 1],
6. ],

dieciséis

1. squareSize: 40,
2. anchura: Ember.computed (**función**() {
3. **devolver este**.obtener('Grid.firstObject.length')
4. }),
5. altura: Ember.computed (**función**() {
6. **devolver este**.obtener('Grid.length');

112

|  |  |
| --- | --- |
| 14: Nivel de salida | 113 |

1. }),
2. PixelWidth: Ember.computed (**función**() {
3. **devolver este**.obtener('anchura')\* **esta**.obtener('SquareSize');
4. }),
5. pixelHeight: Ember.computed (**función**() {
6. **devolver este**.obtener('altura')\* **esta**.obtener('SquareSize');
7. }),

30

1. Esta completo(){
2. **dejar** hasPelletsLeft= **falso**;
3. **dejar** cuadrícula= **esta**.obtener('cuadrícula');
4. grid.forEach ((fila)=>{
5. row.forEach ((celular)=>{
6. **Si**(celda== 2) {
7. hasPelletsLeft = **cierto**
8. }
9. })
10. })
11. **regreso** !hasPelletsLeft;
12. },
13. })

La primera propiedad que ponemos en el nivel es la red.

Esta es la opción más obvia; la red es la representación del diseño del nivel. Si lo cambiamos decuadrícula a diseño (Por lo que estaríamos acceder level.layout en lugar de level.grid) Se haría tan mucho sentido. De hecho, si se sigue el almacenamiento de la cuadrícula en lapac-man componente, que no sería negligente simplemente nombrarlo nivel. Así que, obviamente eso tiene que venir con nosotros.

Entonces traemos en el grupo de funciones que declaran la altura, la anchura, y las versiones de píxeles de los mismos. El venir con ellos es elsquareSizepropiedad que permite traducir de las versiones de la red a las versiones de píxeles de anchura y altura. Esta transferencia también era bastante obvio: que el prefijo todas ellas conpantalla, Pero hubiera funcionado igual de bien para prefijar con nivel. Como lo es ahora, cuando se accede a una de estas propiedades en elpac-man manillar, se llama

level.pixelHeight en lugar de levelPixelHeight.

Ahora, si hubiera tenido un plan maestro para todo esto, lo habría hecho cosas nombrado como levelPixelHeight y diseño en lugar de los semi-sinónimo screenPixelHeight y cuadrícula. pero la suerte *hizo* golpear con el nombre Nivel completado. Es obvio que pertenece a lanivel objeto (otro signo: la única otra cosa que accede es cuadrícula, Que ahora se ha trasladado a la nivelobjeto). Hemos tomado la vuelta y lo renombróEsta completo.

|  |  |
| --- | --- |
| 14: Nivel de salida | 114 |

**Reinicio de él todo**

Dejar de tomar que heurística, hay otra función que se podría pensar debe ser movido sobre: volver a empersar el nivel. Sin embargo, podemos ver que aunque el nombre de método tiene “nivel” en ella, a menudo referencias pac, que la nivelobjeto no tiene acceso a. De hecho, si nos fijamos en elvolver a empersar el nivel función que dará cuenta de que en realidad tiene dos trabajos:

* *//components/pac-man.js*

2 volver a empersar el nivel(){

3*// reiniciar Pac*

4**esta**.conjunto('Pac.x',0);

5**esta**.conjunto('Pac.y',0);

* **esta**.conjunto('Pac.frameCycle',0);
* **esta**.conjunto('Pac.direction','detenido');

8

9*// volver a empersar el nivel*

1. **dejar** cuadrícula= **esta**.obtener('Level.grid');
2. grid.forEach ((fila, rowIndex)=>{
3. row.forEach ((célula, columnIndex)=>{
4. **Si**(celda== 0) {
5. rejilla [rowIndex] [columnIndex] = 2;
6. }
7. });
8. });
9. },

Se añadieron los comentarios para este ejemplo de código, y representan los dos puestos de trabajo que volver a empersar el nivel está haciendo.

Cuando creamos esta función sólo tuvo el segundo trabajo- el de reiniciar el nivel. El primer trabajo, de reiniciar el Pac, se añadió más tarde.

Vamos a seguir adelante y refactorizar esta cambiando el nombre de él y luego aprovechando el hecho de que tenemos objetos de nivel y Pac ahora.

* *//components/pac-man.js*

2 reiniciar(){

* **esta**.obtener('PAC').reiniciar();
* **esta**.obtener('nivel').reiniciar();

5 }

|  |  |
| --- | --- |
| 14: Nivel de salida | 115 |

* *//models/pac.js*

2 reiniciar(){

3**esta**.conjunto('X',0);

4**esta**.conjunto('Y',0);

5**esta**.conjunto('FrameCycle',0);

6**esta**.conjunto('dirección','detenido');

7 }

* *//models/level.js*
* reiniciar(){
* **dejar** cuadrícula= **esta**.obtener('cuadrícula');
* grid.forEach ((fila, rowIndex)=>{
* row.forEach ((célula, columnIndex)=>{

6**Si**(celda== 0) {

7rejilla [rowIndex] [columnIndex] = 2;

* }
* });

10});

11 }

¡Aquí vamos! Todo lo que es propiamente descriptiva, y cada objeto tiene una forma que se reinicie.

**Hacer que todo encaja**

Por supuesto, al igual que lo separa todo lo alto y nombrar las cosas bien no será suficiente para hacer nuestro trabajo aplicación de nuevo. También tendremos que editar algunas partes del código por lo que se hacen referencia a las variables correctas.

Para empezar, la sustitución de todas las referencias a cuadrícula con una referencia a level.grid (Excepto los que están dentro nivel, por supuesto). Luego hacen las sustituciones similares conscreenPixelWidth, screenPixel-Alturay Nivel completado.

Cuando usted está haciendo esto, es posible que observe que el Pacaccede a la rejilla ... pero no tiene acceso a la planta. Vamos a tener que darle acceso a eso.

|  |  |
| --- | --- |
| 14: Nivel de salida | 116 |

* *//components/pac-man.js*

2 didInsertElement () {

3**dejar** nivel=Level.create ()

4**esta**.conjunto('nivel', nivel)

5**dejar** pac=Pac.create ({nivel:nivel})

* **esta**.conjunto('PAC', Pac)
* **esta**.lazo();8 },

Aquí estamos creando un nivel de didInsertElementy se establece en el ámbito de los componentes. Hasta aquí todo bien. Entonces, cuando se crea elPac, Añadimos un hash que tiene sólo un par clave-valor: el ser clave nivel, Y el valor es el nivel que acabamos de crear.

Esto se suma a la que la propiedad Paca objetos, pero sólo para esa instancia del objeto que se está creando actualmente. Si crea otraPac objeto, que será completamente nuevo y no saber nada sobre el nivel que estamos asignando aquí.

Otra propiedad interesante del hash que estamos pasando es que anula cualquier valor que se define en la clase. Para probar eso, tratar de crear instancias con el nivelLevel.create ({squareSize: 20}).

Así que ahora, si hemos hecho todos nuestros movimientos correctamente, el juego debe ser de nuevo en marcha de nuevo- solo con mucho mejor calidad del código.

**LessSharedStuff**

Parte de eso es que la calidad del código SharedStuffahora tiene un enfoque más definido. Si nos fijamos en él, hay dos grupos de código: las cosas relacionadas con el dibujo y el hash-direcciones.

Es tentador querer separar los dos grupos, pero entonces acababa de terminar la mezcla de vuelta en el que tiene direcciones en el que el material de dibujo, ya offsetFor se basa en esas direcciones.

Aquí vamos a hacer un movimiento audaz y decir “mierda”.

Sí, lo sé, se supone que los libros de programación de restar importancia a las piezas de mierda de su código a menos que haya una solución fácil listo, listo para demostrar algún momento, pero no es así como funciona la programación de la vida real. Es decir, hay un montón de minimización de código de mierda en la vida real, pero también hay una gran cantidad de código que es malo, y reconoció a ser malo, y sin embargo, nunca se fija, ya que la fijación no es fácil.

Por lo tanto, en el reconocimiento de la realidad, vamos a dejar una mixin mal definida con un nombre vaga en nuestro código.

Consideré cambiar el nombre del mixin mixins / shitty-name.js o mixins / fuck-it.js, Pero no hay razón para mantener recordando a la gente de este.

Ahora, de vuelta en las mejoras!

|  |  |
| --- | --- |
| 14: Nivel de salida | 117 |

**Otro nivel**

Debido a que hemos separamos nuestra nivel en su propio objeto separado, es trivial para crear un segundo nivel distinto en la parte superior de la primera:

* *//models/level2.js*
* **importar** nivel de'./nivel';

3

4 **por defecto de exportación** Level.extend ({

* squareSize: 60,

6cuadrícula: [

7[2, 2, 2, 2, 2, 2, 2, 2, 2],

8[2, 1, 1, 2, 1, 2, 1, 1, 2],

9[2, 1, 2, 2, 2, 2, 2, 1, 2],

1. [2, 2, 2, 1, 1, 1, 2, 2, 2],
2. [2, 1, 2, 2 ,2, 2, 2, 1, 2],
3. [2, 1, 1, 2, 1, 2, 1, 1, 2],
4. [2, 2, 2, 2, 2, 2, 2, 2, 2],
5. ]
6. })

Estamos importando el nivel, y luego anulando la red con una nueva rejilla de nuestra elección (y el cambio de la squareSize también, sólo por diversión).

Luego, en el pac-man componente que sólo puede sustituir Nivel con Nivel 2:

* *//components/pac-man.js*
* **importar** Nivel 2 de' ../models/level2'

3

4 **por defecto de exportación** Ember.Component.extend (KeyboardShortcuts, SharedStuff, {

5didInsertElement () {

6**dejar** nivel=Level2.create ()

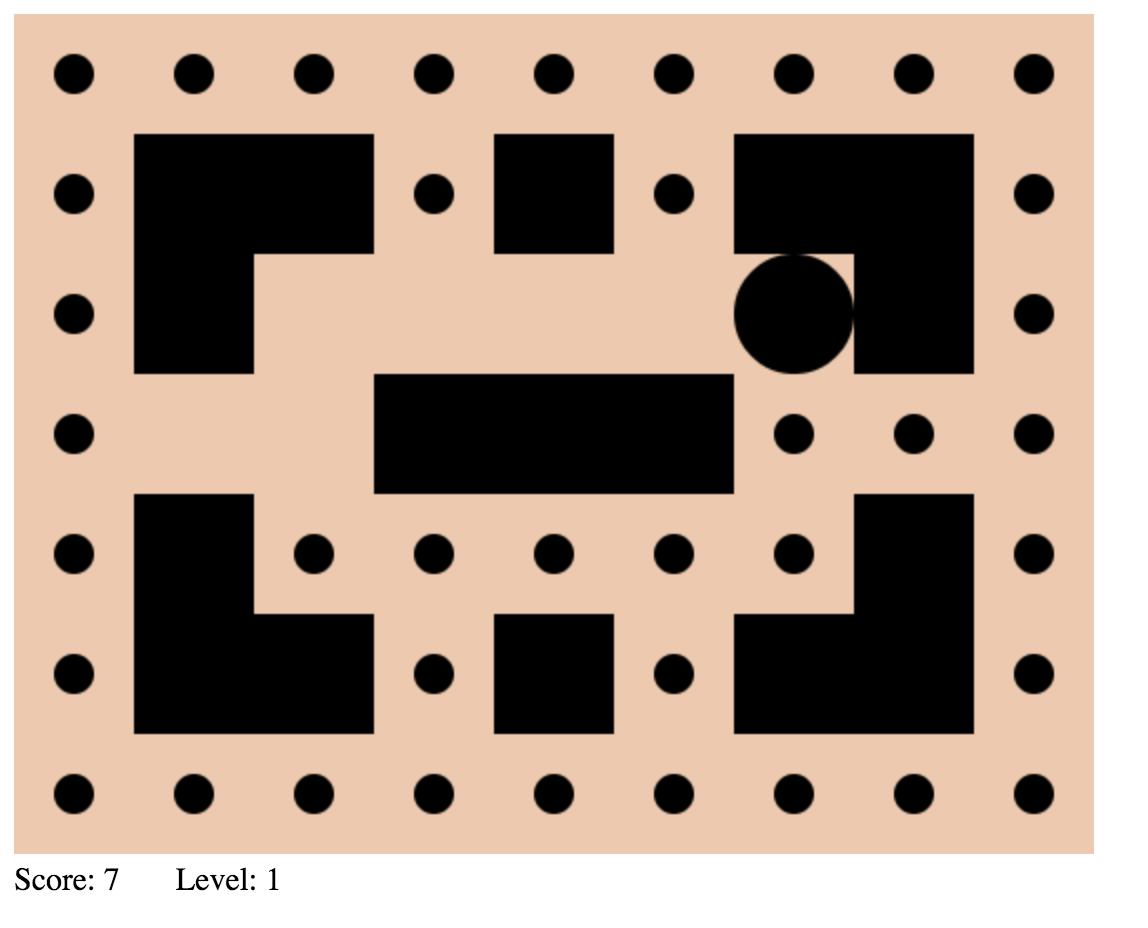
* **esta**.conjunto('nivel', nivel)
* **dejar** pac=Pac.create ({nivel:nivel})

9**esta**.conjunto('PAC', Pac)

1. **esta**.lazo();
2. },
3. ...
4. })

Sólo tiene que importar el nuevo nivel y utilizarlo como la clase creación de instancias y la pluma, el nivel se desconecta. ¡Tan genial!

|  |  |
| --- | --- |
| 14: Nivel de salida | 118 |



Pero hay un problema: nuestra nueva disposición tiene la partida de Pac en una pared. Por suerte ya está entrando en una plaza abierta, pero no siempre puede ser tan afortunado. Hay que especificar la posición de partida del Pac dentro del nivel.

**El Nivel gana más poder regulador**

Vamos a empezar por la creación de un hash en el nivel llamado startingPac:

* *//models/level2.js*

2 startingPac: {

* X: 0,
* y: 3

5 },

a continuación, vamos a utilizar aquellas en las que una instancia de la Pac, Asegurándose de que se inicia en el lugar deseado:

* *//components/pac-man.js*

2 didInsertElement () {

3**dejar** nivel=Level2.create ()

4**esta**.conjunto('nivel', nivel)

5**dejar** pac=Pac.create ({

* nivel: nivel,
* X: level.get ('StartingPac.x'),
* y: level.get ('StartingPac.y')

9})

1. **esta**.conjunto('PAC', Pac)
2. **esta**.lazo();
3. },

|  |  |
| --- | --- |
| 14: Nivel de salida | 119 |

Ya que estamos reemplazando los valores de partida de X y y sobre el Pac, Puede ser que también los puso a nulo. Esto significa que el usuario de la clase que estos valores son para ser aprobada en (vamos a seguir adelante y añadir un valor inicial nulo paranivel mientras estamos en ello).

También nos aseguraremos de que para establecer el X y y los valores correctamente al reiniciar.

* *//models/pac.js*

2 nivel: **nulo**,

3 X: **nulo**,

4 y: **nulo**,

5 reiniciar(){

6**esta**.conjunto('X', **esta**.obtener('Level.startingPac.x'));

7**esta**.conjunto('Y', **esta**.obtener('Level.startingPac.y'));

8**esta**.conjunto('FrameCycle',0);

9**esta**.conjunto('dirección','detenido')

1. },

Y que nos pone a tener siempre la posición correcta, tanto cuando el juego se inicia por primera vez y cuando el juego se reinicia después de una victoria. Si estás pensando en ir de nuevo a su nivel original, asegúrese de darle unastartingPacpicadillo. He utilizado el siguiente:

* *//models/level.js*

2 startingPac: {

* X: 2,
* y: 1

5 },

**Resumen**

En este capítulo continuamos nuestra separación de varios Preocupaciones- esta vez por la separación de nuestro nivel. Además de nuestro código estar mejor organizado y más fácil de leer, por fin empezamos a ver los otros beneficios de la separación de diferentes clases cuando hemos sido capaces de crear rápidamente un segundo nivel. También nos dieron una dosis horrible de la realidad que lo mejor es olvidar.

En el siguiente capítulo vamos por último añadir en nuestros archienemigos del ghosts- y luego vamos a realmente encantados Hemos separado materia hacia fuera!

**En primer lugar que frecuenta**

Era de estudios y de un duro día de Matteo había cambiado el diseño del laberinto, sacudiendo su mapa mental y hacer que se generaliza sus reflejos. Algunas cosas que había aprendido tan útil en el curso pasado eran perjudiciales si se hace sin pensar en este caso, por lo que un conocimiento de la situación comenzó a afianzarse. En la mitad del día se pasó de nuevo, y para el final que tenía una idea aproximada de las ventajas y desventajas de las diferentes técnicas que había aprendido en diferentes situaciones.

Pero a pesar de la dificultad, a pesar de seguir sintiendo el aguijón de la situación con Jerome, a pesar de tener que eliminar una de sus posesiones más preciadas, que fueron capaces de conseguir con él sabiendo que tenía nuevos amigos a ver esta noche.

Usted entra a la sala de oración y lentamente elegir sus amigos desde el desayuno. Es probable que no sería capaz de reconocer a cualquiera de ellos individualmente, excepto para el gran hombre, pero juntos eran distintivo.

El grupo es inusualmente susurrante grave-, acurrucado. Usted se acerca con cautela, pero cuando te ven la totalidad de sus cambios de comportamiento. No es el bullicio de su comportamiento cafetería, pero todavía alegre y acogedor.

“Buen día?” Le pregunte.

“Largo día”, dice uno de ellos.

“Extraño los días”, dice otro.

“No hay que preocuparse acerca de nuestro nuevo amigo, ¿eh?”, Dice el hombre grande. “Un montón de otras cosas para hablar.”

Los otros miran alrededor nervioso, inseguro. Algo les había sacudido. “Usted ve un fantasma o algo?” Le pregunte.

“No sabemos exactamente ... fue ... algo. Algo así como un PAC, pero con estas protuberancias. Estas…"

“tentáculos. Era como tentáculos “.

“Tentáculos?”

“Sí, estas extremidades saltones que algunos animales acuáticos antiguas tenían.”

“De todas formas, tenemos un atisbo de ella. No nos dan demasiada vídeo, porque nos centramos en conseguir el PAC fuera de peligro, pero la sede está analizando lo que tenemos.”

El Imam llama la reunión de oración a la orden, llevando la conversación a un alto. Vas a través de las oraciones, sentirse bien tener hermanos a tu lado una vez más. En los silencios, las lagunas, usted piensa que su propia prayers- los más personales.

120

|  |  |
| --- | --- |
| En primer lugar que frecuenta | 121 |

Todo va bien ... pero luego hay un ruido. Las conversaciones Imam más fuerte, tratando de pensar fuera de la existencia, pero los aumentos en el volumen retumbando aún más, cada vez más difícil de ignorar.

Si el rumor es difícil de ignorar, los gritos es imposible.

Jerome irrumpe a través de la puerta de la sala de oración, algo que en cualquier otra situación sería una ruptura casi imperdonable de la etiqueta. “Salam!” En el momento en que la vuelta él te agarró y se precipita de distancia.

Estás metido en su PAC, esta vez cargado de instrumentos científicos. “Es imperativo que estos objetos permanecen funcional”, dice Jerome. “Usted es el único que entiende la ciencia, pero sabe que necesitará este equipo para tener éxito.”

Jerome mete en un PAC cercano, y se oiga en la radio. "Vamonos."

"¿Dónde?"

"Abajo."

**15: Ghostly**

Los fantasmas fin han llegado! Están aquí para detener sus actividades de pellets de recopilación de una vez por todas! También para hacer el juego más difícil.

El primer paso es crear el fantasma en sí.

**El fantasma de sí mismo**

Vamos a crear una Fantasma clase, a continuación, incluir todo el material necesario con el dibujo SharedStuff mixin.

* *//models/ghost.js*
* **importar** Ember desde'ascua';
* **importar** SharedStuff de'../mixins/shared-stuff';

4

5 **por defecto de exportación** Ascua.Objeto.extend (SharedStuff, {

* dibujar(){
* **dejar** X= **esta**.obtener('X');

8**dejar** y= **esta**.obtener('Y');

9**dejar** radiusDivisor= 2;

1. **esta**.drawCircle (x, y, radiusDivisor, **esta**.obtener('dirección'));
2. },
3. })

La única cosa que estamos definiendo en este momento es la dibujar función, que es igual que el Pac dibujar función.

A continuación vamos a necesitar para conseguir una Fantasmaen nuestra aplicación. Vamos a empezar por la adición a la masa de ejemplificaciones dedidInsertElement:

* *//components/pac-man.js*
* **importar** Ghost desde'../models/ghost';

3

4 didInsertElement () {

* *// ...*
* **dejar** fantasma=Ghost.create ({
* nivel: nivel,
* X: 0,

9y: 0

122

|  |  |
| --- | --- |
| 15: Ghostly | 123 |

1. });
2. **esta**.conjunto('fantasma', fantasma)
3. *// ...*
4. },

Estamos estableciendo el X y y los valores a 0 por defecto en este momento, a pesar de que va a cambiar más adelante.

Luego, durante el bucle vamos a añadir una llamada al fantasma de dibujar función:

* *//components/pac-man.js*

2 lazo(){

* **esta**.obtener('PAC').movimiento();

4

* **esta**.processAnyPellets ();

6

* **esta**.pantalla clara();
* **esta**.drawGrid ();
* **esta**.obtener('PAC').dibujar();

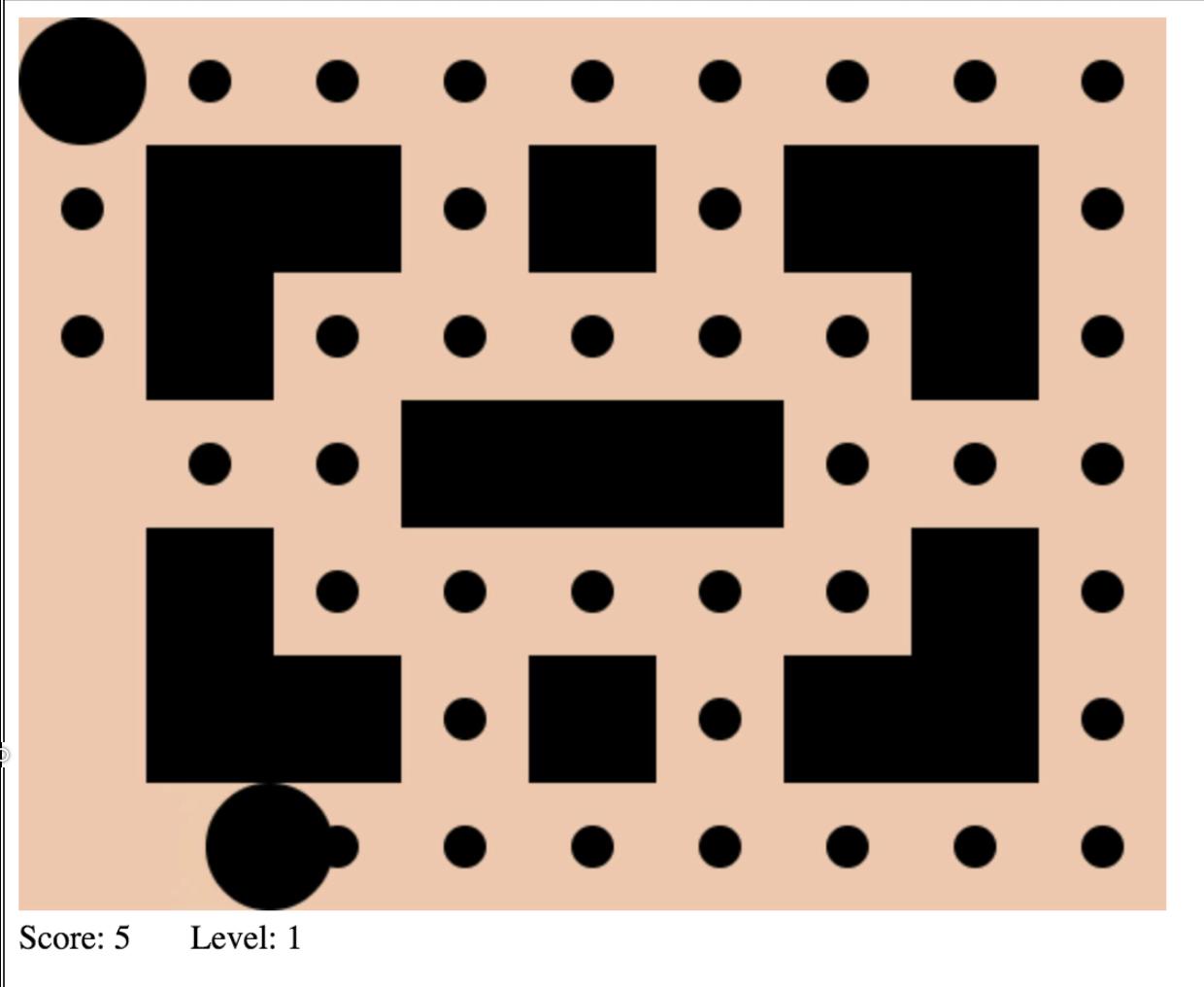
10**esta**.obtener('fantasma').dibujar();

11

12Ember.run.later (**esta**, **esta**.lazo, 1000-1060);

13 },

Todo este código se traducirá en una “Fantasma”Aparecer en la esquina superior izquierda de la pantalla.



Sin embargo, en este momento el “fantasma” sólo se sienta allí. Con el fin de hacer que sea reconocible como un fantasma que necesitaremos añadir:

• Diferenciación de color

|  |  |
| --- | --- |
| 15: Ghostly | 124 |

* Movimiento
* Comportamiento Chasing

Eso es lo que el resto de este capítulo será de alrededor.

**Colores verdaderos**

En primer lugar será la diferenciación del color. El primer cambio será el de nuestra compartidadrawCircle función, addig un quinto argumento opcional con el fin de dar cabida a una mayor flexibilidad:

* *//mixins/shared-stuff.js*
* drawCircle (x, y, radiusDivisor, dirección, color = '# 000') {
* *// ...*
* ctx.fillStyle = color;
* *// ...*

6 },

Por lo que el quinto argumento opcional tiene un valor predeterminado de # 000(negro). Si no hay quinto argumento se pasa en, será negro. Si un quinto argumento se pasa en, entonces tendrá que a medida que el color (anulando el valor por defecto).

Aquí está nuestra función draw renovada para el fantasma, Con el nuevo quinto argumento añadido en.

* *//models/ghost.js*

2 dibujar(){

3**dejar** X= **esta**.obtener('X');

4**dejar** y= **esta**.obtener('Y');

5**dejar** radiusDivisor= 2;

6 **esta**.drawCircle (x, y, radiusDivisor, **esta**.obtener('dirección'), '# F55'); 7 },

Si lo desea, este sería un buen momento para cambiar nuestra Pac al color más amarillo de # FE0.

Nótese que los pellets siguen llamando drawCirclecon sólo 4 argumentos, y siguen siendo negro. Eso significa que todo está funcionando como se esperaba.

**Movimiento: Está vivo!**

Tenemos que hacer nuestra Fantasmamovimiento. Por suerte, ya tenemos un círculo que el moves-Pac. Vamos a extraer un montón de cosas de laPac en un Movimiento mixin:

|  |  |
| --- | --- |
| 15: Ghostly | 125 |

* *//mixins/movement.js*
* **importar** Ember desde'ascua';

3

4 **por defecto de exportación** Ember.Mixin.create ({

5X: **nulo**,

6y: **nulo**,

7nivel: **nulo**,

8dirección: 'abajo',

9

1. movimiento(){...},
2. animationCompleted () {...},
3. finalizeMove () {...},
4. pathBlockedInDirection (dirección) {...},
5. cellTypeInDirection (dirección) {...},
6. nextCoordinate (coordenadas, dirección) {...},
7. })

Todo lo que hemos copiado más ha sido copiado en el pie de la letra. En lugar de ir sobre ellos, uno por uno, puede ser más instructivo mirar por encima de lo que no hemos copiado:

* *//models/pac.js*
* **importar** Ember desde'ascua';
* **importar** SharedStuff de'../mixins/shared-stuff';

4 **importar** movimiento de '../mixins/movement';

5

6 **por defecto de exportación** Ascua.Objeto.extend (SharedStuff, movimiento, {

7dirección: 'abajo',

* intención: 'abajo',

9

1. reiniciar(){...},
2. dibujar(){...},
3. cambia la direccion(){
4. **dejar** intención= **esta**.obtener("intención")
5. **Si**(**esta**.pathBlockedInDirection (intención)) {
6. **esta**.conjunto('dirección','detenido');
7. } **más** {
8. **esta**.conjunto('dirección', La intención);
9. }
10. },
11. })

Hemos dejado intención, reiniciar, dibujary cambia la direccion. También tenemosdirección, Pero eso es anular el dirección que ponemos en Movimiento.

|  |  |
| --- | --- |
| 15: Ghostly | 126 |

salí cambia la direccion ampliado para que se les podría recordar el algoritmo de toma de decisiones utilizado para el Pac. Eso va a ser la principal diferencia entre elPac y el Fantasmas.

Así que vamos a mezclar en Movimiento al Fantasma, Y añadir en un espacio en blanco cambia la direccion por lo que nuestra movimiento función no generará un error:

* *//models/ghost.js*
* **importar** Ember desde'ascua';
* **importar** SharedStuff de'../mixins/shared-stuff';

4 **importar** movimiento de '../mixins/movement';

5

6 **por defecto de exportación** Ascua.Objeto.extend (SharedStuff, movimiento, {

* dibujar(){...}
* cambia la direccion(){
* *// Lo decidiremos más adelante*

10}

11 })

El único que queda por hacer ahora es añadir el movimiento al bucle.

* *//components/pac-man.js*
* **importar** Ghost desde'../models/ghost';

3

4 lazo(){

* **esta**.obtener('PAC').movimiento();
* **esta**.obtener('fantasma').movimiento();

7

* **esta**.processAnyPellets ();

9

1. **esta**.pantalla clara();
2. **esta**.drawGrid ();
3. **esta**.obtener('PAC').dibujar();
4. **esta**.obtener('fantasma').dibujar();
5. Ember.run.later (**esta**, **esta**.lazo, 1000-1060);
6. },

¡Se está moviendo! ¡Hurra!

Por desgracia, sólo se mueve hacia abajo y luego cae fuera de la pantalla. Es evidente que nuestro Santo va a necesitar algunas habilidades para tomar decisiones mejores si se quiere atrapar a nuestros héroes.

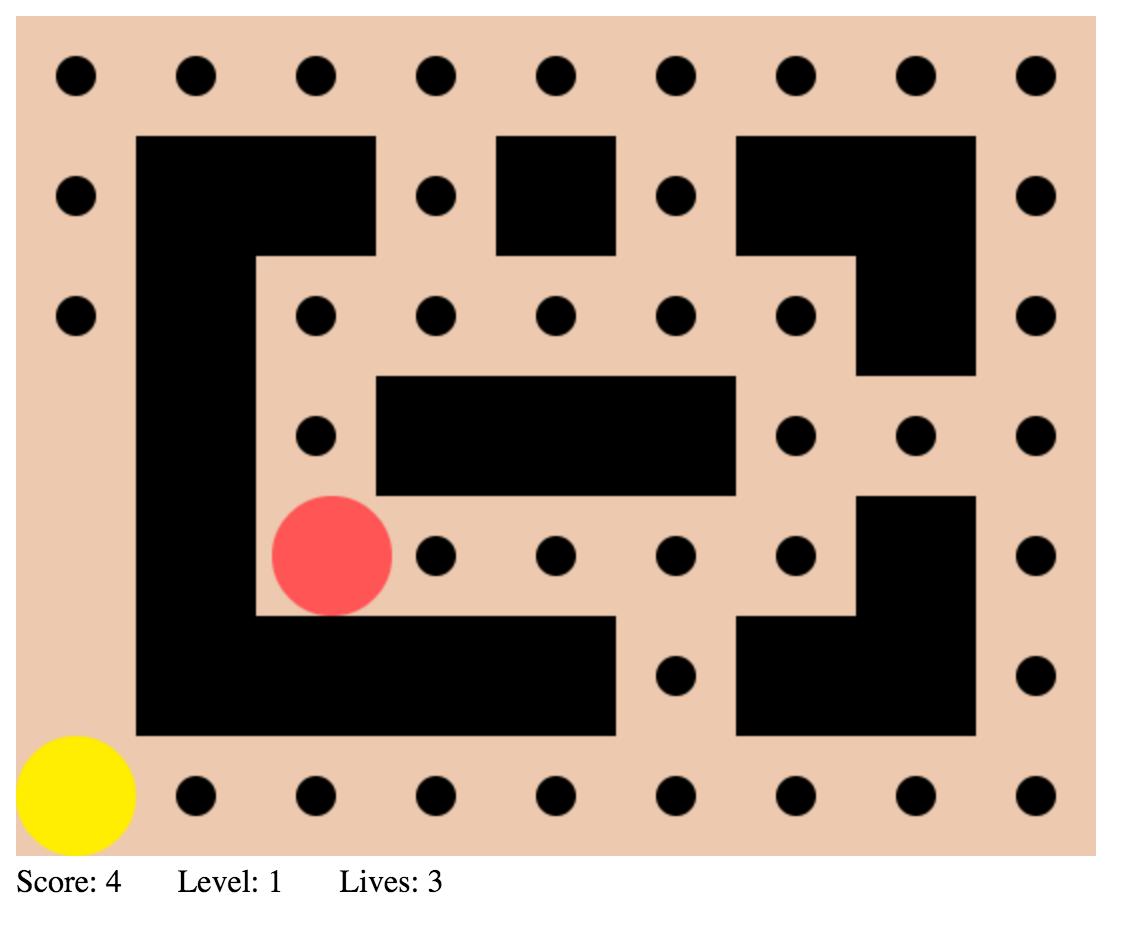
|  |  |
| --- | --- |
| 15: Ghostly | 127 |

**Los fantasmas de calor de búsqueda**

Aquí es donde definimos cambia la direccionpara nuestro Santo. Por desgracia, no es tan fácil como simplemente tomar direcciones del usuario. Programaremos una IA muy primitivo, uno que combina las capacidades de búsqueda de calor con un poco de aleatoriedad.

El calor de búsqueda, simplemente se ve a donde el Pac es y trata de hacer que el fantasma de movimiento en esa dirección, ya sea acercándose horizontal o verticalmente, lo que sea más conveniente.

La aleatoriedad es allí por dos razones. La primera es por lo que es más difícil conseguir un fantasma atrapado en una esquina, ya que a menudo haría en un algoritmo de búsqueda de calor puramente determinista.



En la situación anterior, un fantasma puramente determinista nunca alcanzará el PAC. Mientras que eso puede hacer que para algunas situaciones interesantes rompecabezas (véase la eventual secuela de este libro, EDITAR PACMAN (2017?)), No es tan grande para el juego. Por lo tanto, la aleatoriedad.

La segunda razón de la aleatoriedad es para que cuando añadimos varios fantasmas, que no convergen en una stack- una vez que llegan en la misma plaza una vez, entonces, por definición, siempre se mueven en la misma “mejor” dirección y ser, por todos los efectos, un fantasma. Así, una vez más, no es muy bueno para el juego. Por lo tanto, la aleatoriedad.

Muy bien, así que vamos a construir esta cosa:

|  |  |
| --- | --- |
| 15: Ghostly | 128 |

* cambia la direccion(){
* **dejar** direcciones=['izquierda','derecho','arriba','abajo']
* **dejar** directionWeights=directions.map ((dirección)=>{

4**devolver este**.chanceOfPacmanIfInDirection (dirección);

* })

6

7**dejar** bestDirection= **esta**.getRandomItem (direcciones, directionWeights);

8**esta**.conjunto('dirección', BestDirection)

9 },

A medida que avanzamos más cambia la direccion, Hay tres cosas aquí que lo más probable es que no reconoce:

1. los mapa Método (integrado en Javascript).
2. los chanceOfPacmanIfInDirection función.
3. los getRandomItem función.

No se preocupe, vamos a explicar todos- a partir de la función común e increíblemente útil Javascript mapa.

**Mapa**

mapa es un método de hermanos a para cada, Que hemos usado antes. Ambos trabajan en matrices, y hacer algo con cada miembro de la matriz. La diferencia es que mientraspara cada está destinado a realizar una acción en cada elemento de una matriz (como dibujo una célula forEach pared que es igual a 1), mapa se entiende para devolver una nueva matriz transformada, y la función dentro de mapa es lo que hace el mapeo de la matriz de origen a la nueva matriz.

Vamos a dar un ejemplo sencillo:

* **dejar** frutas=['Cereza','fresa','naranja']

2 **dejar** yummyFruits = fruits.map ((fruta)=>{

3**regreso** `delicioso **ps**Fruta**}**`;

4 });

5 console.log (yummyFruits) *// [ 'cereza delicioso', 'fresa delicioso', 'naranja delicioso']*

mapa se llama en el frutas matriz y toma una función. El primer argumento de esta función es un elemento en la matriz (el segundo elemento es el índice, pero no estamos utilizando eso aquí). A continuación, anteponer la palabra “delicioso” de la fruta, y que volvemos nuevo valor.

Veremos que el uso paralelo en nuestra cambia la direccion método:

|  |  |
| --- | --- |
| 15: Ghostly | 129 |

* **dejar** direcciones=['izquierda','derecho','arriba','abajo']
* **dejar** directionWeights=directions.map ((dirección)=>{

3**devolver este**.chanceOfPacmanIfInDirection (dirección);

4 })

En este caso, en lugar de simplemente anteponiendo la palabra 'delicioso', estamos llamando a un método que calcula la probabilidad de que el fantasma se encuentra el Pac yendo en esa dirección.

Veamos que el método siguiente.

**¿Dónde está el Pac?**

Como veremos pronto, el nombre chanceOfPacmanIfInDirectionno es precisamente correcto, pero es lo suficientemente cerca. Vamos a profundizar en.

* chanceOfPacmanIfInDirection (dirección) {
* **Si**(**esta**.pathBlockedInDirection (dirección)) {

3**regreso** 0;

4} **más** {

5 **dejar** posibilidades = ((**esta**.obtener('Pac.y') - **esta**.obtener('Y')) \* **esta**.obtener(`direcciones.**ps**\ 6 dirección**}**.y`)) +

7 ((**esta**.obtener('Pac.x') - **esta**.obtener('X')) \* **esta**.obtener(`direcciones.**ps**\

* dirección**}**.x`))
* **regreso** Mates.MAX (posibilidades,0)+ 0,2

10}

11 },

En primer lugar, vamos a comprobar para ver si el camino está bloqueado en esa dirección. Si es así, volvemos a 0. De lo contrario, hacemos un cálculo más complicado.

La primera línea de la posibilidades cálculo es el siguiente:

* (**esta**.obtener('Pac.y') - **esta**.obtener('Y')) \* **esta**.obtener(`direcciones.**ps**dirección**}**.y`)

La primera parte está recibiendo la diferencia entre el y valor del fantasma y la yvalor de la Pac-que puede ser positivo o negativo (o 0, si están en la misma fila). A continuación, lo multiplicamos por que la dirección deyvalor. Por lo tanto 'arriba' terminaría siendo-1 y 'derecho' terminaría siendo 0.

Por lo tanto 'derecho' e 'izquierda' terminará automáticamente como un 0 grande, no importa la diferente yvalores. Luego de 'arriba' y 'abajo' es -1 o 1. Esa señal es muy importante- si es el mismo signo que la diferencia entre el PAC y el fantasma dey valores, entonces posibilidades el resultado final será positivo.

O, dicho de otro modo, si el Pac es en esa dirección, entonces el resultado será positiva y cuanto más lejos se encuentra en esa dirección, cuanto mayor sea el resultado positivo será. Si el Pac es en la dirección opuesta, el resultado será negativo.

Pero no es sólo el y de valor que añadimos a ese resultado una operación similar para X valores:

|  |  |
| --- | --- |
| 15: Ghostly | 130 |

* **dejar** posibilidades=((**esta**.obtener('Pac.y')- **esta**.obtener('Y'))\* **esta**.obtener(`direcciones.**ps**terrible\

2 cción**}**.y`)) +

3 ((**esta**.obtener('Pac.x') - **esta**.obtener('X')) \* **esta**.obtener(`direcciones.**ps**terrible\

* cción**}**.x`))

Esa segunda línea, con el X valores, lo hace de 'izquierda' y 'derecha' lo que la primera línea hizo por 'arriba' y 'abajo'.

Entonces el posibilidadesterminará con algún tipo de valor, por lo general positivo o negativo. A continuación, redondeamos todo valor negativo y añadir un poco de aleatoriedad adicional:

* **regreso** Mates.MAX (posibilidades,0)+ 0,2

los Math.max la llamada se encuentra el mayor de posibilidades y 0, esencialmente, redondeando a todos los valores negativos

1. La adición de 0,2 hace que sea posible que el fantasma a veces va a ir la dirección opuesta de la Pac ... aunque no muy a menudo. El cambio de esta variable puede cambiar como juega el juego.

Así que esa es la forma de calcular las “posibilidades” de la Pac siendo en cualquier dirección.

**El (Weighted Random) de Chase**

A continuación vamos a volver y reorientar a nosotros mismos en nuestra más grande cambia la direccion función:

* cambia la direccion(){
* **dejar** direcciones=['izquierda','derecho','arriba','abajo']
* **dejar** directionWeights=directions.map ((dirección)=>{

4**devolver este**.chanceOfPacmanIfInDirection (dirección);

* })

6

7**dejar** bestDirection= **esta**.getRandomItem (direcciones, directionWeights);

8**esta**.conjunto('dirección', BestDirection)

9 },

Acabamos directionWeights desde el mapa, Y es una matriz tal como [0, 3,2, 0,2, 0,2]. Eso significa que no hay ninguna posibilidad de ir a la izquierda (probablemente una pared allí), una gran oportunidad de subir, y una pequeña posibilidad de ir hacia abajo o derecha. Esto se llama “aleatoriedad ponderada”.

Ahora sólo tenemos una función que se llevará a esta matriz y elegir una dirección al azar, pero preferencing las direcciones con los pesos más grandes. Esa función esgetRandomItem:

Tenga en cuenta que algo falsamente enviar el resultado de getRandomItem a bestDi-rection. ¿Es realmente el “mejor” dirección, si es que lo seleccionamos semi-aleatoria? Considéralo nuestra probablyPrettyGoodDirection.

|  |  |
| --- | --- |
| 15: Ghostly | 131 |

* getRandomItem (lista, peso) {
* **var** peso total=weight.reduce (**función** (Prev, cur, i, arr) {

3**regreso** prev+canalla;

* });

5

6**var** random\_num=Mates.aleatorio()\*peso total;

7**var** weight\_sum= 0;

8

9**para** (**var** yo= 0; yo<List.length; yo++) {

1. weight\_sum + = peso [i];
2. weight\_sum = Número(Weight\_sum.toFixed (2));
3. **Si** (random\_num<weight\_sum) {
4. **regreso** lista [i];
5. }
6. }
7. },

santa madre de Pac, que es un montón de cosas que no hemos visto antes. Y algunos de ellos (comoreducir) Son temas avanzados. Estoy declarando que la comprensión de esta función está fuera del alcance de este libro-sólo copiar y pegar en el código y saber que sí aleatoriedad ponderado.

Por lo que la función toma en las direcciones y la matriz de pesos, y se escupe una cadena de una dirección.

Esa cadena se establece como la nueva dirección.

Y hay que ir- tenemos un algoritmo para ir después de la Pac. Pero antes de que pueda hacer eso, tenemos que hacer una última cosa: darle de comer una versión del Pac.

* *//components/pac-man.js*
* **importar** Ghost desde'../models/ghost';

3 **importar** pac a partir '../models/pac';

4

5 didInsertElement () {

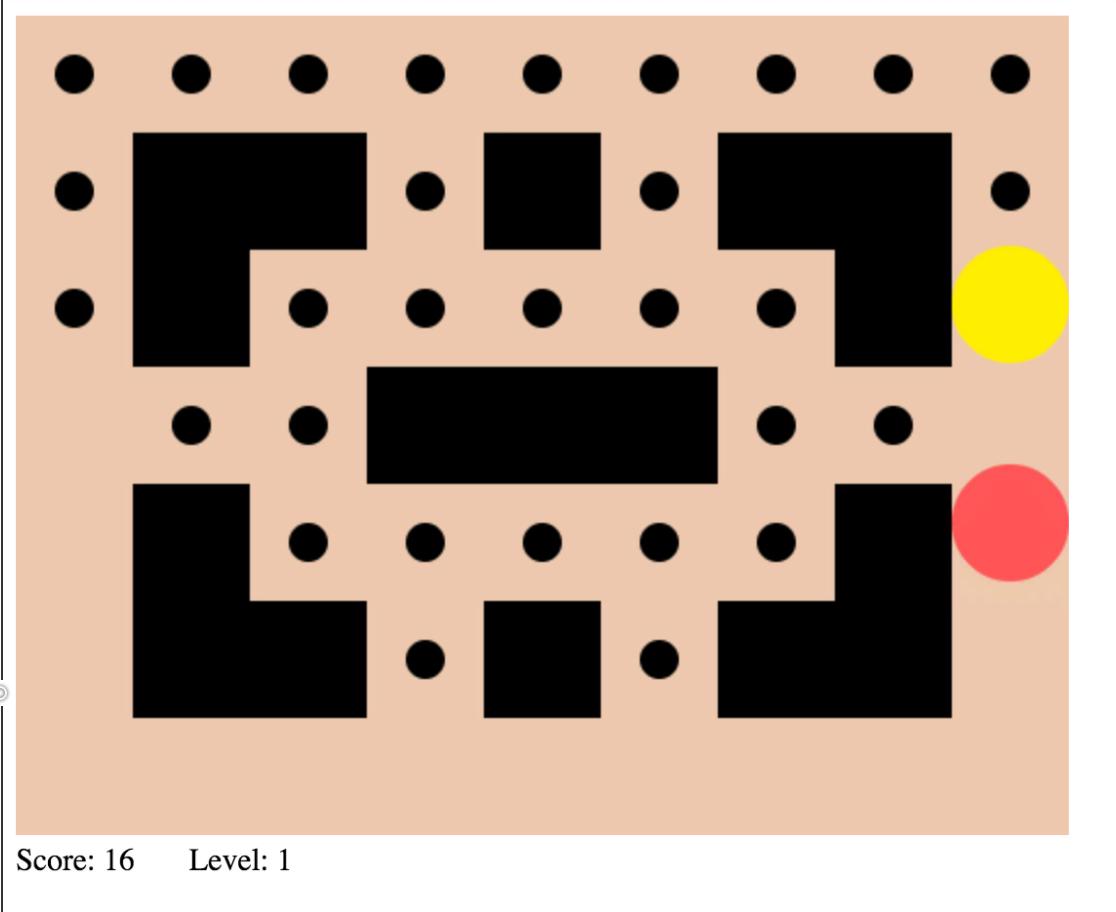
* *// ...*
* **dejar** pac=Pac.create ({...})
* *// ...*
* **dejar** fantasma=Ghost.create ({

1. nivel: nivel,
2. X: 0,
3. y: 0,
4. pac: pac
5. });
6. **esta**.conjunto('fantasma', fantasma)

|  |  |
| --- | --- |
| 15: Ghostly | 132 |

1. *// ...*
2. },

Esto completa nuestro trabajo sobre el Ghost AI, pero no hay mucho espacio para mejorar en este sentido. En el Pacman original, cada uno de los 4 fantasmas tenía su propia “personalidad” AI que hace que se comporte un poco diferente que los otros fantasmas. Eso sería un PR frío para dar a[la addon¹²](https://github.com/jeffreybiles/pac-man)!



**Resumen**

En este capítulo hemos creado una Fantasmaclase y la instancia en nuestro componente. Para ello, se sacó un montón de código de laPac clase y lo puso en el Movimientomixin, que a continuación incluimos en ambos. A continuación, creó un algoritmo para hacer que nuestraFantasma buscar la Pac, Proporcionando por algún modo de juego mucho más emocionante.

En el camino hemos aprendido acerca de la mapa función, que es vital para trabajar eficazmente con matrices.

En el siguiente capítulo, vamos a añadir en consecuencias para el funcionamiento en una detección de colisiones Ghost- de averiguar si estamos Hit, una función de reinicio para cuando recibe un golpe, y una vida contraria a recordarnos de nuestra mortalidad. En el capítulo después de eso vamos a aumentar el número de fantasmas y es más fácil para establecer sus lugares de partida en un nivel.

¹²<https://github.com/jeffreybiles/pac-man>

**Sacrificio**

El Santo da abajo en usted, una bola de furia malévola. Tentáculos golpeando contra las paredes, catapultar hacia delante y violentamente de dirección a través de las esquinas. Se pone de pie horrorizado por un momento hasta golpear su instinto. No hay igualdad de muerto, no luchar contra ella. Vuelo es lo único que queda.

PAC se mueve con la misma agilidad, por suerte no registrar el temblor de las manos. todo su cuerpo.

“Izquierda”, dice Jerome sobre el auricular. Gire a la izquierda en la próxima oportunidad.

"Derecho. Ahora a la derecha. Recta de nuevo. Derecho. Derecha. Derecho. Izquierda. Mierda, no, quiero decir recta. Oh, mierda."

Es demasiado tarde. Que ya ha girado a la izquierda, y pronto se une a él en su cursing- que es un callejón completamente muerto. Se mira atrás y ve el fantasma viene en la esquina, es batida violenta llegar a una parada. Se sabe que usted tiene. Tentáculos onda, a la deriva, como marchas a su vez. Se habla más de los auriculares, pero no se puede oír el gemido que viene del Santo. Los gemidos se detiene, y los engranajes comienzan a girar con mayor velocidad.

“Ya lo tengo”, dice una voz. Es el hombre grande. Usted sólo lo conoció ayer, pero ya se le considera uno de sus mejores amigos aquí.

Los engranajes del fantasma abierta, revelando algo que se ve ... humano.

“Allahu Akbar!” Grita el hombre grande como su PAC se estrella contra el fantasma. El impulso empuja ambas máquinas fuera de su camino, y una frenética Jerome está gritando a que tome ventaja de esto y se mueve.

La figura humana del fantasma se dejó caer contra la pared, pronunciando algo. ¿Crees que se oye su nombre de algún sitio, pero eso no puede ser correcto.

El propio Santo ha entrado en modo de autodefensa. Dos tentáculos se trituran, provocando inútilmente. Los que aún funcionan están destrozando PAC del hombre grande.

“Move!” Grita Jerome. "¡Venga!"

Usted sabe lo suficiente sobre la arquitectura PAC saber qué tan cerca están los tentáculos de carne.

Están cerca.

Están ahí.

Uno se pregunta cuánto tiempo va a recordar esta imagen.

133

**16: Contacto**

En el capítulo anterior hemos creado un fantasma que persiguió al Pac, pero cuando se llamó la Pac, no pasó nada. Eso no es un juego divertido! El no debe tener consecuencias!

Y en este capítulo, prevalecerán las consecuencias! Pero primero, vamos el reto al aumentar el número de fantasmas.

**múltiples fantasmas**

La adición de la segunda fantasma es muy fácil. Simplemente copia el código que crea nuestro primer fantasma, y ​​luego cambiamos el valor que asignamos a:

* *// en didInsertElement en /components/pac-man.js*

2 **dejar** Ghost1 = Ghost.create ({

3nivel: nivel,

* X: 0,
* y: 0,
* pac: pac 7 });

8 **dejar** Ghost2 = Ghost.create ({

9nivel: nivel,

1. X: 5,
2. y: 0,
3. pac: pac
4. })

Nos hemos puesto 5 espacios a la derecha de modo que podemos verlo como un fantasma derecho separado en la puerta de salida.

La necesitamos para asegurarse de que el bucle hace que tanto los Fantasmas movimientos. Lo haremos por primera asignándolos a lafantasmas array, a continuación, un bucle sobre la matriz.

134

|  |  |
| --- | --- |
| 16: Contacto | 135 |

* *// en /components/pac-man.js*

2

3 didInsertElement () {

* *// ...*
* **dejar** fantasmas=[Ghost1, Ghost2]
* **esta**.conjunto('fantasmas', Fantasmas)
* *// ...*

8 }

9

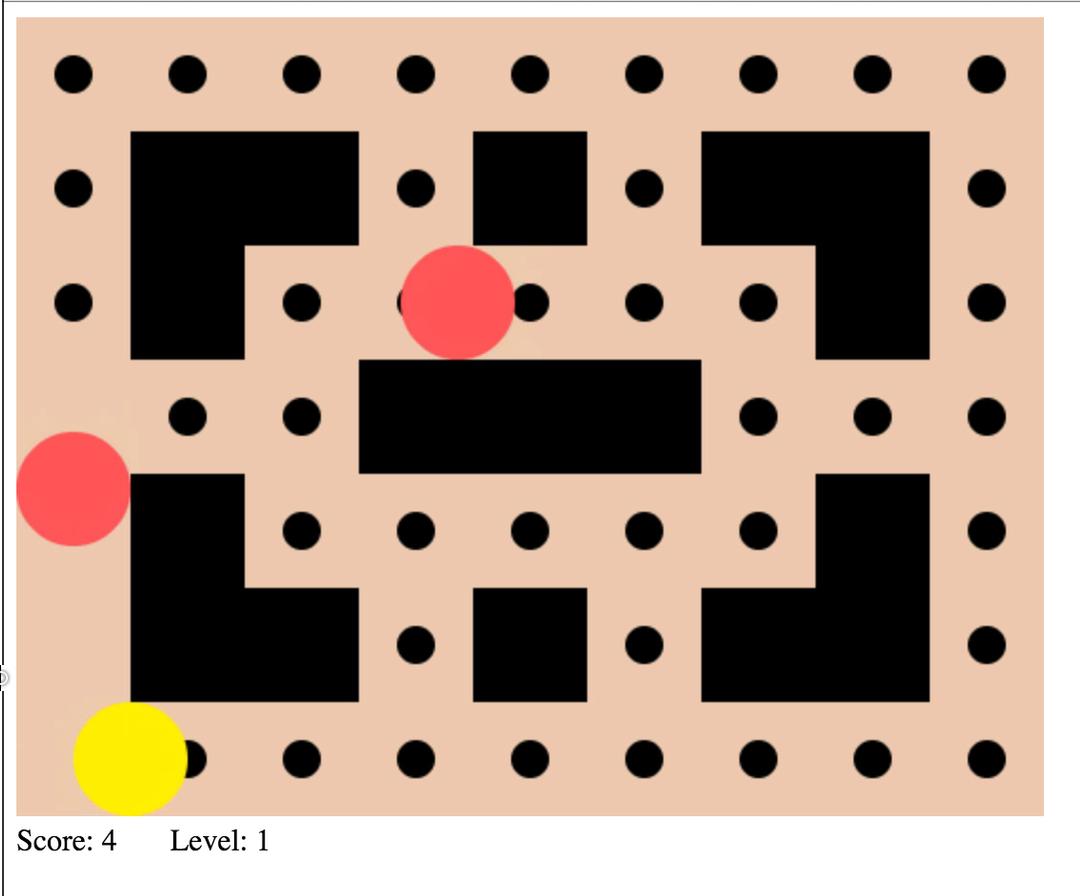
1. lazo(){
2. **esta**.obtener('fantasmas') .forEach ((fantasma)=>{Ghost.move ()});
3. *// ...*
4. }

Esto puede parecer como un trabajo extra en este momento- por qué no acaba de establecer por separado y luego llamar movimiento en dos de ellos con la mano? -, pero que dará sus frutos más tarde, cuando tenemos más fantasmas y más métodos que queremos hacer un llamamiento a todos ellos.

Más tarde, vamos a añadir en el código siguiente en el dibujar método:

* *// en didInsertElement en /components/pac-man.js*
* **esta**.obtener('fantasmas') .forEach ((fantasma)=>{Ghost.draw ()})

Con esto, estaremos bucle y dibujo ambos fantasmas. Si quisiéramos añadir una tercera, que sólo había necesidad de crear un tercer fantasma endidInsertElement y luego incluir que en el fantasmasformación. Todo el resto del código permanecería igual.



Antes de continuar, quería introducir algunos matices y accesos directos a nuestra sintaxis de la función de flecha.

|  |  |
| --- | --- |
| 16: Contacto | 136 |

**Funciones mejores Flecha**

En los dos últimos ejemplos de código, hemos estado usando la función de flecha de la siguiente manera:

* (fantasma)=> {Ghost.draw ()}

Sin embargo, hay algunos atajos que podemos tomar. En primer lugar, si sólo hay un argumento, entonces no son necesarios los paréntesis. En segundo lugar, si sólo hay una expresión en la función, no son necesarias las llaves. El código anterior podría tener el siguiente aspecto:

* fantasma => ghost.draw ()

Observe cómo mucho más limpio que es! Y no hay información relevante se perdió. Aquí está dentro del contexto del código:

* *// en `loop`*
* **esta**.obtener('fantasmas') .forEach (fantasma=>ghost.move ());

3 *// ...*

4 **esta**.obtener('fantasmas') .forEach (fantasma => ghost.draw ());

Una cosa a tener en cuenta cuando se está utilizando esto es que si se agrega una segunda expresión (digamos que desea llamar ghost.draw () y el método hipotético ghost.sound ()), Entonces usted necesita para volver a agregar las llaves. Y si se añade un segundo argumento, es necesario volver a añadir los paréntesis. Sin embargo, para este tipo de situaciones, este acceso directo hace que nuestro código mejor!

Ahora, de vuelta a la construcción del juego! A continuación vamos a poner en el código que puede decir si nos hemos quedado en un fantasma.

**Detección de colisiones**

Estamos en primer lugar va a crear una función que puede decir si un fantasma se ha topado con nuestra Pac. Lo haremos mediante la comprobación para ver si comparten la mismaX y y valores (es decir, que están en la misma plaza).

* collidedWithGhost () {
* **devolver este**.obtener('fantasmas') .any ((fantasma)=>{
* **devolver este**.obtener('Pac.x')==ghost.get ('X')&&

4 **esta**.obtener('Pac.y')==ghost.get ('Y')

* })

6 },

|  |  |
| --- | --- |
| 16: Contacto | 137 |

Así se está haciendo la fantasmasmatriz, y luego llama al cualquier método, que hemos usado antes. Se ejecuta una función para cada elemento de la matriz, y si alguno de ellos regresan cierto, entonces laalgunala función devuelve verdadero. En este caso, sólo estamos comprobando si elX y y valores del fantasma en cuestión coincide con la X y y valora nuestra Pac.

A continuación, vamos a llamar a que dentro de nuestro bucle, y reiniciar todo si es cierto.

* lazo(){
* **esta**.obtener('PAC').movimiento();
* **esta**.obtener('fantasmas') .forEach (fantasma=>ghost.move ());

4

* **esta**.processAnyPellets ();

6

* **esta**.pantalla clara();
* **esta**.drawGrid ();
* **esta**.obtener('PAC').dibujar();

1. **esta**.obtener('fantasmas') .forEach (fantasma=>ghost.draw ());
2. **Si**(**esta**.collidedWithGhost ()) {
3. **esta**.reiniciar();
4. }

15

1. Ember.run.later (**esta**, **esta**.lazo, 1000-1060);
2. },

Entonces tendremos que editar nuestro reiniciar función para incluir nuestros fantasmas.

* reiniciar(){
* **esta**.obtener('PAC').reiniciar();
* **esta**.obtener('nivel').reiniciar();
* **esta**.obtener('fantasmas') .forEach (fantasma=>ghost.restart ());5 },

Si se ejecuta el código de ahora, se verá que todo está trabajando estupendo, los pellets llegar a poner en su posición, el PAC va a donde originalmente fue colocado, y los fantasmas ... bueno, ambos van en la esquina superior izquierda , una encima de la otra.

Eso es porque, aunque nos propusimos las posiciones de los fantasmas de manera diferente cuando los creamos, al restablecer los pone en ellas [0, 0]. Lo que necesitamos es una posición de inicio predeterminada para los fantasmas definidos en el nivel, al igual que hemos definido el punto de partida de la Pac en el nivel.

|  |  |
| --- | --- |
| 16: Contacto | 138 |

**Lugares, por favor**

Vamos a empezar con la fabricación de sólo un pequeño cambio que lleva a cabo nuestro objetivo para este nivel específico.

A continuación, vamos a generify el arreglo de modo que los niveles son una vez más intercambiables.

El cambio inicial se lleva a cabo dentro de la Fantasma clase:

* *//models/ghost.js*

2 en eso() {

3**esta**.conjunto('StartingX', **esta**.obtener('X'));

4**esta**.conjunto('StartingY', **esta**.obtener('Y'));

5**devolver este**.\_super (... argumentos);

6 },

7 reiniciar(){

8**esta**.conjunto('X', **esta**.obtener('StartingX'));

9**esta**.conjunto('Y', **esta**.obtener('StartingY'));

1. **esta**.conjunto('FrameCycle',0);
2. **esta**.conjunto('dirección','detenido')
3. },

En primer lugar tenemos nuestro en esofunción. Es similar adidInsertElement en el que se ejecuta al comienzo del ciclo de vida de un objeto, pero al mismo tiempo didInsertElement sólo se puede utilizar en los Componentes y otras cosas que heredan de Vista, en eso se puede utilizar en cualquier cosa que se hereda de Ember objeto (es decir, la mayoría de las cosas).

Mientras didInsertElement se ejecuta cuando se inserta el elemento en el DOM, en esose ejecuta cuando se inicializa el objeto. Los componentes se insertan en el DOM, pero la mayoría de los objetos se no- por lo tanto ellos no ser capaz de utilizardidInsertElement. A pesar de nuestros objetos tienen unadibujar función, que está pintando una representación en el lienzo, no insertar un elemento en el DOM.

Por lo que establecer nuevos valores startingX y startingY dentro de en eso función, utilizando los valores que damos por X y y. Cuando llamamosreiniciar, Que utiliza esos valores para restablecer X y y.

La tercera línea en en eso le puede desconcertar al principio- vamos a explicar por qué lo hacemos en esta sección próxima opcional.

**Súper Spread (opcional)**

La siguiente parte es más avanzado, y no es necesario para entender el resto del libro

|  |  |
| --- | --- |
| 16: Contacto | 139 |

Al crear una instancia de un objeto, en esose ejecute. Cualquier objeto en la cadena objeto puede definiren eso. Esto puede ser un problema porque una definición que es más baja en la cadena de herencia anulará una definición que es más arriba. Si definimosen eso, Y algo más arriba también se define en eso e hizo cosas importantes allí, vamos a romper cosas con nuestra en eso definición.

Ahí es donde \_súper() entra en acción. Se envía el comando de la cadena, diciendo: “A pesar de que he definido en eso aquí abajo, quiero correr ningún en esos que están por encima de mí en la cadena de herencia”. A continuación, se alimenta en ... argumentos.

Hemos utilizado previamente ...para señalar que “estamos dejando fuera parte del código”, como se usa en Inglés normal. Sin embargo, aquí tiene un significado muy específico. Es el operador de “difundir”, que toma una matriz y se extiende hacia fuera.

* **dejar** nums=[1,2,3];
* console.log ([nums, 4, 5]) *// [[1, 2, 3], 4, 5]*

3 (console.log [... nums, 4, 5]) *// [1, 2, 3, 4, 5]* 4

5 **dejar** argumentos = ['FirstArg', 'SecondArg']

6 **esta**.\_super (argumentos) *//this.\_super(['firstArg', 'secondArg'])*

7 **esta**.\_super (... arguments) *//this.\_super('firstArg', 'secondArg')*

Así que en lugar de utilizar la matriz, es el uso de cada elemento por separado de la matriz que está siendo difundido.

En este caso, argumentos es una matriz de JavaScript que se crea automáticamente para cada función que contiene cada argumento dado a la función cuando se está llamando.

Eso es todo por la parte opcional. Fue breve, avanzó, y tal vez no lo entendía todo. ¡Esta bien! Ahora, de vuelta a nuestra programación regular.

**Nivel sabe mejor**

Ahora queremos hacerlo de modo que podemos tener diferentes posiciones de partida fantasma en diferentes niveles.

Así es como lo definimos en el nivel 2:

|  |  |
| --- | --- |
| 16: Contacto | 140 |

* *// modelos / nivel2*

2 startingPac: {

* X: 0,
* y: 3

5 },

6 startingGhosts: [{

* X: 0,
* y: 0

9 }, {

1. X: 5,
2. y: 0
3. }],

He dejado el startingPac allí para referencia, para que pueda ver que estamos haciendo básicamente lo mismo, simplemente pegando múltiples de ellos en una matriz.

A continuación, vamos a querer crear esos fantasmas en didInsertElement:

* *// componentes / PAC-man.js*

2 didInsertElement () {

* *// ...*
* **dejar** fantasmas=level.get ('' startingGhosts) .map ((startingPosition)=>{

5**regreso** Ghost.create ({

6nivel: nivel,

7X: startingPosition.x,

8y: startingPosition.y,

* pac: pac

1. })
2. })
3. **esta**.conjunto('fantasmas', Fantasmas)
4. *// ...*
5. }

Nótese que debido a que estamos usando mapa en la matriz, podemos crear tantos fantasmas como el nivel especifica.

Y entonces ... eso es todo! Antes de pasar a dar a nuestros múltiples vidas Pac, asegúrese de agregar posiciones fantasma de partida para level1 y cualquier otro nivel que ha hecho.

**vidas extra**

La parte final de este capítulo es el más simple; una vuelta de la victoria agradable para todo su trabajo duro.

Vamos a empezar por la adición de la vive alojamiento hasta el pac-man componente:

|  |  |
| --- | --- |
| 16: Contacto | 141 |

* *//components/pac-man.js*

2 vive: 3

A continuación vamos a mostrar en el manillar para ese componente:

* *<! - templates / componentes / PAC-man.hbs ->*
* <**lona** carné de identidad="myCanvas" anchura={{}} Level.pixelWidth altura={{}} Level.pixelHeight></ \

3 canvas>

4 <**br**>

5 Puntuación: {{score}} **& Nbsp; & Nbsp; & Nbsp;** Nivel: {{}} levelNumber **& Nbsp; & Nbsp;** ynbsp \

6 ; Vidas: {{}} vidas

Eso hará que los niveles de aparecer debajo de nuestro nivel. Ahora necesitamos para el Pac perder una vida cuando se realiza un fantasma:

* *//components/pac-man.js*

2 lazo(){

* *// ...*
* **Si**(**esta**.collidedWithGhost ()) {
* **esta**.decrementProperty ('vive');
* **esta**.reiniciar();
* }

8

9Ember.run.later (**esta**, **esta**.lazo, 1000-1060);

1. },

Así que cada vez que chocan con un fantasma, se pierde una vida antes de reiniciar el nivel.

¿Qué pasa cuando se llega a 0 vidas?

* *//components/pac-man.js*

2 reiniciar(){

3**Si**(**esta**.obtener('vive')<= 0) {

* **esta**.conjunto('Puntuación',0)
* **esta**.conjunto('vive',3)
* **esta**.obtener('nivel').reiniciar();
* }
* **esta**.obtener('PAC').reiniciar();
* **esta**.obtener('fantasmas') .forEach (fantasma=>ghost.restart ());10 },

|  |  |
| --- | --- |
| 16: Contacto | 142 |

Añadimos la Sideclaración que sólo se activa cuando el jugador está fuera de la vida, y luego restablecer la puntuación y el número de vidas. También cambiamos la llamada reinicio de niveles en allí, de modo que no se reiniciará el nivel cuando se muere a menos que esté fuera de la vida.

Esto quiere decir que tendremos que reiniciar el nivel específicamente cuando queremos reiniciar cosas debido al nivel que se completó:

* *//components/pac-man.js*

2 processAnyPellets () {

3**dejar** X= **esta**.obtener('Pac.x');

4**dejar** y= **esta**.obtener('Pac.y');

5**dejar** cuadrícula= **esta**.obtener('Level.grid');

6

7**Si**(GRID [y] [x]== 2) {

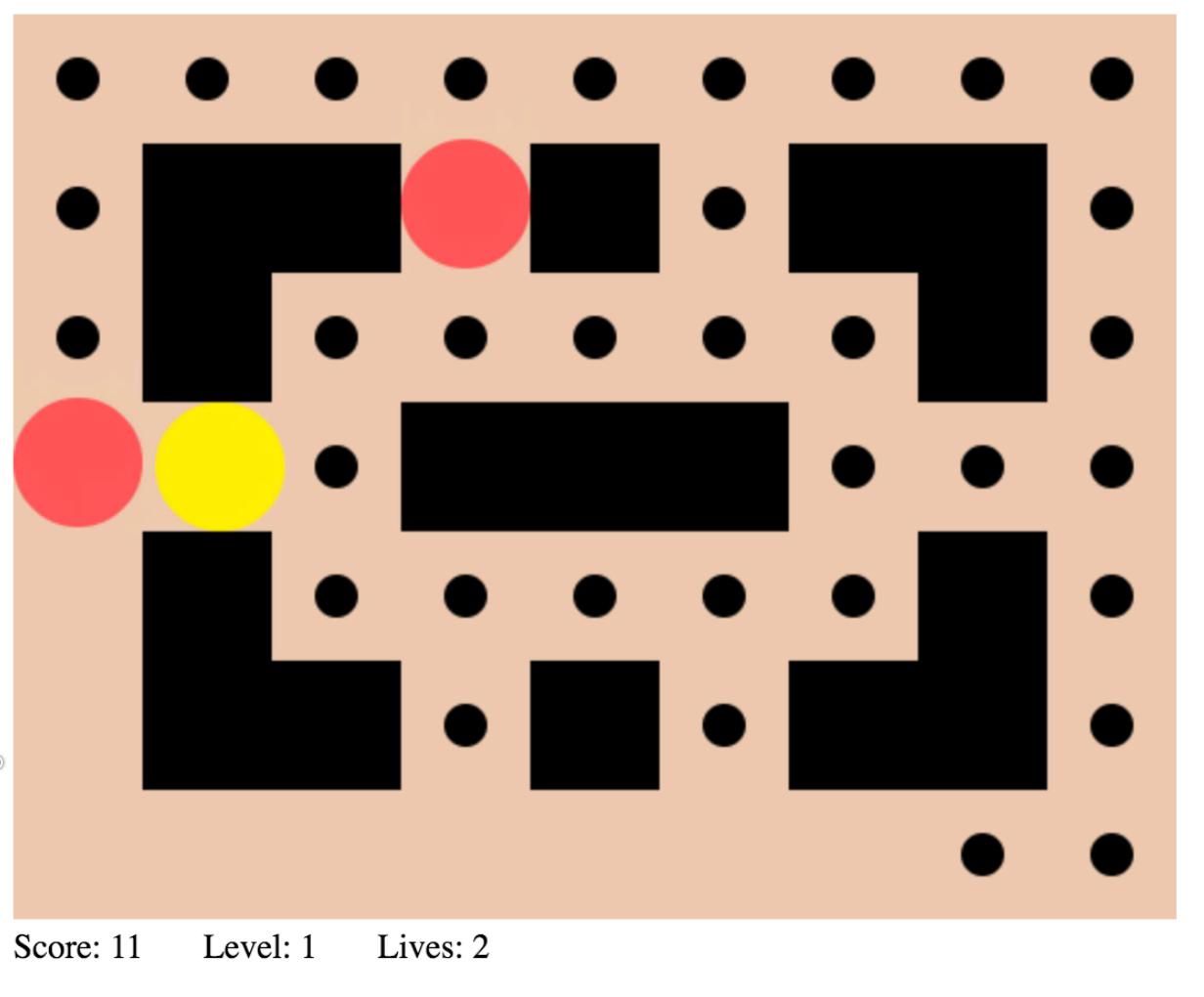
8rejilla [y] [x] = 0;

* **esta**.incrementProperty ('Puntuación')

10

1. **Si**(**esta**.obtener('nivel').Esta completo()){
2. **esta**.incrementProperty ('LevelNumber')
3. **esta**.obtener('nivel').reiniciar();
4. **esta**.reiniciar()
5. }
6. }
7. },

Por lo que termina a cabo nuestra característica de vidas extra.



|  |  |
| --- | --- |
| 16: Contacto | 143 |

**Resumen**

En este capítulo hemos logrado mucho: hemos añadido varios fantasmas, hicimos los fantasmas peligrosos, dio nuestras vidas múltiples Pac, y permitió que cada nivel para determinar cuántos fantasmas que tiene y dónde deben comenzar. También aprendimos una nueva versión más corta de la sintaxis de la función de flecha, así como algunas cosas acerca opcional\_súper y el operador de difusión.

En el siguiente capítulo vamos a hacerlo de modo que ganar un nivel en el que se mueve a un nivel diferente. Sus jugadores van a estar ahogando en variedad!

**Restos**

Es sólo horas después de la fuga que Jerome te deja descansar. Es pasada la medianoche, y la adrenalina está empezando a desaparecer.

"Treinta minutos. Eso es todo. Entonces nos movemos “.

Tienes que salir de la PAC, agarra un pellet de sus tiendas, y lo puso en el procesador. Se zumba, las reacciones químicas y cuidadosamente calibrado calor y presión haciendo que el pellet en algo comestible.

Se toma un momento para mirar alrededor y descubrir patrones extraños en los túneles.

"¿Qué es este lugar?"

“Una extraña ciudad desde el viejo mundo. Debe su nombre a un hombre 'Ahl al-Kitab santa que luchó por los pobres, pero fue tomado por una serie de cultos que practicó la vida salvaje y licenciosa. Ellos fueron útiles en su tiempo, pero al final no aceptarían el Califato, así que tuvimos que destruirlos.”

“Los mató a todos?”

Jerome negó con la cabeza. "No. Hubo algunos que se presenten. Y el 'Ahl al-Kitab, puede estar seguro, se les permitió vivir como dhimmi. Pero los cultos eran terco, vile- algunos se dedicaron a obtener riqueza y poder sin medida, haciéndose dioses en la tierra, mientras que la gente común de hambre. Uno de sus sectas fueron volando en el cielo antes de que pudiéramos detenerlos, pero que destruyeron el resto. Luego hubo otra dedicada a las abominaciones de Sodoma y Gomorra. Era el placer a la lluvia sobre ellos brimstones duro como arcilla cocida, difusión, capa sobre capa “.

Hubiera sabido vagos contornos de esta historia, pero al escuchar la historia sin rodeos le hace mal del estómago. Es evidente que sus antepasados ​​eran uno de los que se convirtieron y se salvaron (sólo más tarde que llegaron a ser apóstata), pero eso no lo hace más fácil de tomar. Esta vez había sido su civilización.

Pero, como su abuelo había dicho, todos los grandes imperios se construyó sobre atrocidades. El Califato había destruido esta civilización, pero dejó más de la mitad de su población viva. Esta civilización muerta casi había aniquilado a toda la población de los que habían estado allí antes que ellos. Y antes de eso, incluso al principio, una especie llamada los neandertales.

Los golpes de procesador, y Jerome te entrega su porción de la pastilla. No está seguro de que tienes hambre más, pero de todos modos se come. La comida toma el borde de un poco de su irritabilidad, resulta algo de él en curiosidad.

“Este azufre ... que es por eso que es tan profundo bajo tierra? Literalmente, enterramos?”Se nota que usted todavía está utilizando‘nosotros’, a pesar de que Jerome sabe lo que eres. Los viejos hábitos son difíciles, y quién sabe quién más podría estar escuchando?

“El azufre es figurativo, una referencia a una historia más atrás. Se utilizó cohetes, bombas, pistolas, cuchillos. El soterramiento vino después. Cuando la comida se acabó y empezó a cavar, la suciedad de la

144

|  |  |
| --- | --- |
| Restos | 145 |

túneles tenían que ir a alguna parte. Eventualmente, todos los grandes ciudades- incluso los que submitted- tuvo que ser enterrado por lo que podría alimentar a nuestra gente “.

Nos fijamos en las paredes, tratando de uncrush en tu mente y recrear lo que había estado allí antes.

Se exasperantemente difícil, después de haber visto sólo unos escasos fotos de estas antiguas ciudades por encima del suelo.

“Usted ha estudiado mucho en esto, ¿no es así.”

“Es fascinante”, dijo. “Es bueno saber tu pasado. Lo siento por lo que gran parte de ella es mi pueblo son pollas a su gente “.

Te encoges. Su vientre se está llenando, y todo esto parece distante ahora, con la amenaza fantasma y todo. “Por lo que sé, estábamos pollas a que a veces también.”

"Sí. Pero es la dicking final que todo el mundo recuerda. La mayor parte de los vencedores recuerdo como gloriosa, pero los que estudian la historia son algo más desgarrado “.

“Me sorprende que los ancianos les permite todo esta investigación.”

Jerome se encoge de hombros. “El gran utilidad como nosotros tienen ciertas libertades que los demás no se conceden. No es suficiente libertad para convertirse en apóstata, pero suficiente libertad para investigar sujetos que son Haram “.

Jerome se levanta, y seguir su ejemplo.

"¿Más abajo?"

"Más abajo. Para las bolas de poder “.

**17: Moving On Down**

La persecución es situ tenemos Pacs huyendo de los fantasmas, quedar atrapado, corriendo a través de sus reservas de vida. Las cosas se ven bastante sombrío para los Pacs! Sin embargo, en este y en los siguientes capítulos vamos a darles un par de herramientas para cambiar el rumbo.

La primera es para hacerles pasar a otro nivel. Decisiones es que ya no sólo tiene que repetir el mismo nivel una y otra hace que su tarea parezca un poco menos de Sísifo. Por supuesto, todavía van a recorrer todos los niveles disponibles (sin pantalla para que usted ganar!), Pero es menos evidente.

**La matriz de Nivel**

Vamos a empezar por la definición de la matriz de nivel:

* *// componentes / PAC-man.js*
* **importar** nivel de'../models/level'

3 **importar** Nivel 2 de ' ../models/level2'

4

5 **por defecto de exportación** Ember.Component.extend (*/ \* \* Mixins /*, {

6niveles: [Nivel, Nivel 2],

* *// ...*

8 })

Estamos importando los dos niveles que hemos definido (se puede definir e importar más si lo desea) y ponerlos en una matriz. Ahora vamos a utilizar esa matriz para seleccionar cuál de ellos para empezar, en lugar de la codificación dura de uno u otro como estábamos antes.

Vamos a definir el método que vamos a utilizar para hacer eso (niveles gama incluye como referencia):

* *// componentes / PAC-man.js*

2 niveles: [Nivel, Nivel 2],

3 loadNewLevel () {

4**dejar** levelIndex=(**esta**.obtener('LevelNumber')- 1)% **esta**.obtener('Levels.length')

5**dejar** levelClass= **esta**.obtener('niveles') [LevelIndex]

6**regreso** levelClass.create ()

7 }

146

|  |  |
| --- | --- |
| 17: Moving On Down | 147 |

Este método termina la sustitución de la Level.create () o Level2.create () llamamos estábamos usando antes.

La primera línea está tomando el levelNumber (Que comienza a las 1 y luego sube en 1 cada vez que vencer a un nivel) y convertirlo en una herramienta para encontrar la correcta Nivel. Primero restamos 1, ya que las matrices (como elniveles array) están indexados comenzando en 0 en lugar de a partir de 1. A continuación, utilizamos el operador resto (%) Con el fin de asegurarse de que estamos lanzó un nivel real. Así que si tenemos 2 niveles, ylevelNumber es 3, sería volver a la primera ((3 - 1)% 2 == 0).

La segunda línea detiene levelIndex y lo utiliza para buscar la clase de nivel de la niveles formación.

En la tercera línea vamos a crear una instancia de la clase. Recuerde, en los niveles de picadillo no estamos almacenar cadenas, estamos almacenar una referencia a la misma clase real, por lo que es vivo y podemos crear instancias de ella.

Volvemos el resultado de esa creación.

¿Cómo es la utilizada en otros lugares? Bueno, podríamos usarlo simplemente para reemplazarLevel2.create () dentro didInsertElement.

* *// componentes / PAC-man.js*

2 didInsertElement () {

3**dejar** nivel= **esta**.loadNewLevel ();

* *// ...*

5 }

Eso es un gran comienzo, pero tenemos planes más grandes.

**Los niveles cambiantes**

Actualmente, hay exactamente un lugar en el código donde queremos cargar un nuevo nivel y sus concomitantes Pacs, pellets, y fantasmas. Cuando el juego ha terminado, el hecho de perder o ganar, simplemente reiniciar. Cambio del nivel que complica.

El conjunto del aparato que tenemos en didInsertElement(Aparte de iniciar el bucle), pasará a llamarse tanto al inicio del juego y cuando cambiamos niveles. Por lo tanto, hay que desacoplar el código dedidInsertElement.

|  |  |
| --- | --- |
| 17: Moving On Down | 148 |

* *// componentes / PAC-man.js*

2 didInsertElement () {

* **esta**.startNewLevel ();
* **esta**.lazo();5 }, 6

7 startNewLevel () {

8**dejar** nivel= **esta**.loadNewLevel ();

9level.restart ()

1. **esta**.conjunto('nivel', nivel)
2. **dejar** pac=Pac.create ({
3. nivel: nivel,
4. X: level.get ('StartingPac.x'),
5. y: level.get ('StartingPac.y')
6. });
7. **esta**.conjunto('PAC', Pac);

18

1. **dejar** fantasmas=level.get ('' startingGhosts) .map ((startingPosition)=>{
2. **regreso** Ghost.create ({
3. nivel: nivel,
4. X: startingPosition.x,
5. y: startingPosition.y,
6. pac: pac
7. })
8. })
9. **esta**.conjunto('fantasmas', Fantasmas)
10. },

Se trata de una gran cantidad de código, pero es más que nada teniendo el código de didInsertElement (Aparte de iniciar el bucle) y se pega en startNewLevel. Entonces llamamosstartNewLevel desde dentro didInsertElement, Lograr el mismo efecto, pero ahora lo que permite el código abstracto que se llama de otra parte.

Tenga en cuenta que no importa cuántas veces empezamos un nuevo nivel, que sólo estamos empezando el bucle de una vez. Eso es porque a partir de múltiples veces es una receta para el mal funcionamiento y / o extraño comportamiento: mejor mantener el mismo bucle y simplemente seguir adelante con nuevas versiones del Pac, el nivel, y los fantasmas.

Hay otro cambio, y que está llamando level.restart (). Usted puede pensar: “¿Por qué estamos llamando reinicio en un nivel de nueva creación? Sin duda, no necesitamos hacer eso!”. Y para subir de nivel es muy cierto ... pero por alguna razón, cuando derrote el primer nivel pero luego quedas sin vida en el siguiente nivel, el juego me reinicia en un nivel 1 que está totalmente desprovisto de pellets. No sé por qué sucede esto, pero esta llamada lo arregla.

|  |  |
| --- | --- |
| 17: Moving On Down | 149 |

Este interludio va a mostrar que a menudo es más fácil de solucionar un problema que para entenderlo. Utilizar su mejor juicio acerca de cuándo es suficiente.

Así que ahora sólo podemos llamar startNewLevel de otra parte.

* processAnyPellets () {
* **dejar** X= **esta**.obtener('Pac.x');

3**dejar** y= **esta**.obtener('Pac.y');

4**dejar** cuadrícula= **esta**.obtener('Level.grid');

5

6**Si**(GRID [y] [x]== 2) {

7rejilla [y] [x] = 0;

* **esta**.incrementProperty ('Puntuación')

9

1. **Si**(**esta**.obtener('nivel').Esta completo()){
2. **esta**.incrementProperty ('LevelNumber')
3. **esta**.startNewLevel ()
4. }
5. }
6. },
7. reiniciar(){
8. **Si**(**esta**.obtener('vive')<= 0) {
9. **esta**.conjunto('Puntuación',0)
10. **esta**.conjunto('vive',3)
11. **esta**.conjunto('LevelNumber',1)
12. **esta**.startNewLevel ()
13. }
14. **esta**.obtener('PAC').reiniciar();
15. **esta**.obtener('fantasmas') .forEach (fantasma=>ghost.restart ());
16. },

Cosas geniales ! De aquí en adelante, puede divertirse creando nuevos niveles y ponerlos en diferentes orders- es casi como que tiene un juego completo!

Bono: Actualmente estamos estableciendo el Pac y Ghost posiciones de partida desde el nivel ... usted podría encontrar la manera de especificar otras variables, como el color o la velocidad (framesPerMove-ment).

**Resumen**

En este capítulo hemos cubierto cómo crear una progresión nivel. ¡Muy genial! Y como un tributo a lo poderoso que nuestro conocimiento Javascript y Ember se ha convertido, sólo tuvimos que aprender una nueva construcción (el operador resto%) Con el fin de hacerlo bien.

|  |  |
| --- | --- |
| 17: Moving On Down | 150 |

Por supuesto, aprender cosas nuevas siempre es buena, no me malinterpreten. Pero ser capaz de codificar algo sin levantar la vista de nuevas construcciones es como ser capaz de mantener una conversación sin tener que aprender una nueva palabra: muy práctico.

Ahora estamos en la recta final, y el resto del libro sólo será puesta en características interesantes como teletransportadores y pellets de energía. ¡Disfruta el viaje!

**Patrulla**

No está seguro de cómo, pero los fantasmas han alcanzado. Cada nivel se baja parece haber vagando en busca de usted. Como tal, y Jerome ha acumulado un repertorio de trucos para detectar y evitar dichas prácticas al tiempo que el buceo profundo y recoger el máximo número de gránulos.

Noche de tiempo (ahora aún menos conectado con el ciclo solar de lo que era antes) se turnan vigilando. La parte más difícil no es el sueño reducido, es saber que en cualquier momento usted podría ser sacado de su descanso y tienen que hacer su escape. La parte más aterradora, como el agotamiento se establece, se pregunta si será capaz de levantarse a tiempo para el próximo ataque.

Sin embargo, continúa, a la baja. Patrones se repiten.

De vez en cuando Jerome se encuentra un dispositivo de comunicaciones que le permite hablar con la gente encima de la tapa. Terrance ha sobrevivido, está escondido, pero Matteo fue muerto en la primera ola. No se pregunta acerca de otras personas por su nombre, pero el número de víctimas es asombroso. Ellos insisten en que hay que encontrar las bolas de poder y hacer un arma.

Finalmente llegar a una nueva pieza de tecno- un teletransportador.

“Los pellets de aquí abajo son más densas, que nos permite construir estas cosas. Cuando vas en una sola, que le lleva a otra parte del mapa “.

Los teletransportadores son útiles, pero aún mejor es el hecho de que no parece que los fantasmas de entender cómo funcionan. Usted agarra algunos muy necesario descanso por acampar justo en el borde de un teletransportador y simplemente hacer el salto cada vez que un fantasma aparece cerca. Es una estrategia arriesgada sin embargo, si los fantasmas nunca averiguar el sistema de teletransporte, así que después de tres días de hacer esto Jerome pone en jaque a la misma.

Y más abajo que vaya. Los días y las noches se confundían.

La pantalla PAC dice que han pasado dos semanas.

151

**18: Teleport**

Teleportation- en este caso, la capacidad de saltar de un cuadrado en un extremo del nivel de un cuadrado correspondiente en el otro extremo de la nivel- suena como una característica que será difícil, pero en realidad es bastante fácil. No sólo eso, pero es fácil de encender y apagar basado en el nivel- que significa que si usted diseñó un montón de niveles después del último capítulo, no hay necesidad de volver atrás y volver a escribir.

**Encontrar Qué Cambiar**

Vamos a echar un vistazo a la situación actual de nuestra tecnología:

* *//mixins/movement.js*
* pathBlockedInDirection (dirección) {
* **dejar** cellTypeInDirection= **esta**.cellTypeInDirection (dirección);
* **regreso** Ember.isEmpty (cellTypeInDirection)||cellTypeInDirection=== 1;5 }, 6

7 cellTypeInDirection (dirección) {

8**dejar** nextX= **esta**.nextCoordinate ('X', Dirección);

9**dejar** nextY= **esta**.nextCoordinate ('Y', Dirección);

10

1. **devolver este**.obtener(`Level.grid.**ps**nextY**}**.**ps**nextX**}**`);
2. },

13

1. nextCoordinate (coordenadas, dirección) {
2. **devolver este**.get (coordenadas)+ **esta**.obtener(`direcciones.**ps**dirección**}**.**ps**coordinar**}**`\
3. );
4. },

Estos métodos, que se mezclan en tanto a la Pac y los fantasmas, nos están diciendo qué direcciones se nos permite entrar a vivir. pathBlockedInDirection toma una dirección, y luego encuentra el tipo de célula en esa dirección (a través cellTypeInDirection). Si se trata de una pared o es nula, entonces se vuelve cierto- el camino está bloqueado de hecho.

Ese valor nulo no está allí debido a un error- que está ahí porque representa cuando cellTypeInDi-rection 'Cae fuera de la red'. Se ve en la parrilla y devuelve un valor nulo para esas coordenadas PORQUE no existen esas coordenadas. Esas son las paredes invisibles alrededor de los niveles que hemos dado por hecho. Cuando añadimos el teletransporte, que volveremos otra cosa en lugar de null.

152

|  |  |
| --- | --- |
| 18: Teleport | 153 |

Para obtener el tipo de células, que llega a la red, pero los valores de X y que llegue a la red con son determinados por nextCoordinate.

nextCoordinate Actualmente es muy simple: se calcula una coordenada (X o Y) a la vez, diciendo lo que coordinan sería si nos cambiamos un cuadrado en la dirección pasó de entrada. Hace esto tomando el valor actual de que coordinar, a continuación, añadir el cambio que se produciría en base a la dirección de pasado-in y el hash direcciones (ya sea 0, -1, o 1).

finalizeMove, Lo que hace el movimiento real, también se hace un llamamiento a nextCoordinate. Eso significa que cambiando nextCoordinate no sólo cambiará la forma en que determinamos si podemos pasar o no, que va a cambiar la forma en que se hace realmente el movimiento, y lo hará de una manera que son coherentes entre sí.

**A dónde vamos desde aquí**

Así que hemos encontrado el método que se necesita cambiar- vamos a llegar a cambiar!

* nextCoordinate (coordenadas, dirección) {
* **dejar** siguiente= **esta**.get (coordenadas)+ **esta**.obtener(`direcciones.**ps**dirección**}**.**ps**COORDINA\3 Te**}**`);
* **Si**(**esta**.obtener('Level.teleport')) {
* **Si**(dirección=='arriba'||dirección=='abajo') {
* **devolver este**.modulo (al lado, **esta**.obtener('Level.height'))

7} **más** {

8**devolver este**.modulo (al lado, **esta**.obtener('Level.width'))

* }

1. } **más** {
2. **regreso** siguiente;
3. }
4. },

14

1. modulo (num, mod) {
2. **regreso** ((Num+mOD)%mod);
3. },

Vamos a empezar con lo que es familiar: la primera línea de nextCoordinate. Es el mismo cálculo, pero en lugar de volver directamente se asigna asiguiente. Ahora bien, si el valor detelepuerto en el nivel cargado en ese momento es falsa, volverá siguiente- en esencia que nos da el mismo método que antes. Pero si el valor es cierto, hacemos algo diferente.

En primer lugar vamos a dividir en dos ramas diferentes en función de la dirección que se pasa. Si es hacia arriba o hacia abajo, vamos a la primera rama en la que va a utilizar la altura del nivel. Si se trata de izquierda o derecha, vamos a la segunda rama en la que va a utilizar el ancho de nivel.

|  |  |
| --- | --- |
| 18: Teleport | 154 |

A continuación, vamos a utilizar nuestra propia función módulo especializado. Así que si hacemos un módulo regular (%), Funciona muy bien para las cosas que son más altos que el divisor. 5% 2 == 1. Pero cuando se llega a los números negativos, el módulo no se convierte automáticamente en ellas positivas. Asi que-1% 2 == -1. Si queremos que el Pac ir a la parte superior del nivel y se muestran en la parte inferior, tendremos que arreglar eso. Por lo tanto nuestra propia función módulo, que sólo se suma en el divisor antes de ejecutar módulo integrado de Javascript.

Mirando hacia atrás en nextCoordinate con este conocimiento, vemos que estamos dando el siguiente valor de la coordenada y de módulo-ción con la dimensión nivel que corresponda.

Como ejemplo: si se va fuera de la derecha de la pantalla de un nivel de ancho de 8, entonces la siguiente valor será de 8, y (recuerda: todo esto es la base de cero, por lo que 7 es el último valor disponible para un nivel con un ancho de 8) nuestro módulo va a llevarlo a un valor de 0. Entonces, si te das la vuelta, su siguiente valor será -1, y nuestro módulo va a llevarlo a un valor de 7.

Así que esa es la forma de hacer un teletransportador.

Mientras tanto el Pac y los fantasmas serán capaces de teletransportarse, no es común en absoluto para los fantasmas teletransportarse. Esto se debe a su capacidad de buscadores de calor no toman en cuenta el teletransporte, por lo que casi siempre se moverá en la dirección opuesta. Recuerde que en el diseño de niveles.

**ENCENDERLO**

Para utilizar la función de teletransporte, usted tiene que establecer teletransporte: trueen su nivel. He aquí un ejemplo:

* **importar** Ember desde'ascua';

2

3 **por defecto de exportación** Ascua.Objeto.ampliar({

4telepuerto: **cierto**,

* *// ...*

6 })

Si lo establece así en el nivel de base, será automáticamente cierto en los niveles secundarios a menos que active específicamente lo apaga en ellos. Si no se establece en un nivel en absoluto, y no está heredando de un nivel que lo tiene establecido, entonces se volverá automáticamentenulo. nulo termina actuando de la misma como falso.

**Un Teleport-Primer Nivel**

Aunque podemos activar el teletransporte en cualquier nivel, se siente mal tener toda una línea de cuadrados teleportable. Se siente más adecuado para un nivel de teletransporte a estar rodeado de paredes, excepto en ciertos puntos que son cortadas para revelar un telepuerto.

Aquí muestro un nivel que se parece más a lo que normalmente se ve como un nivel de teletransporte. Note la casi sólida1'S en el exterior- esas son las paredes que enmarcan el nivel y hacen que las plazas de teletransporte se sientan especiales.

|  |  |
| --- | --- |
| 18: Teleport | 155 |

* *//models/teleport-level.js*

2 **importar** nivel de './nivel';

3

4 **por defecto de exportación** Level.extend ({

* squareSize: 60,

6cuadrícula: [

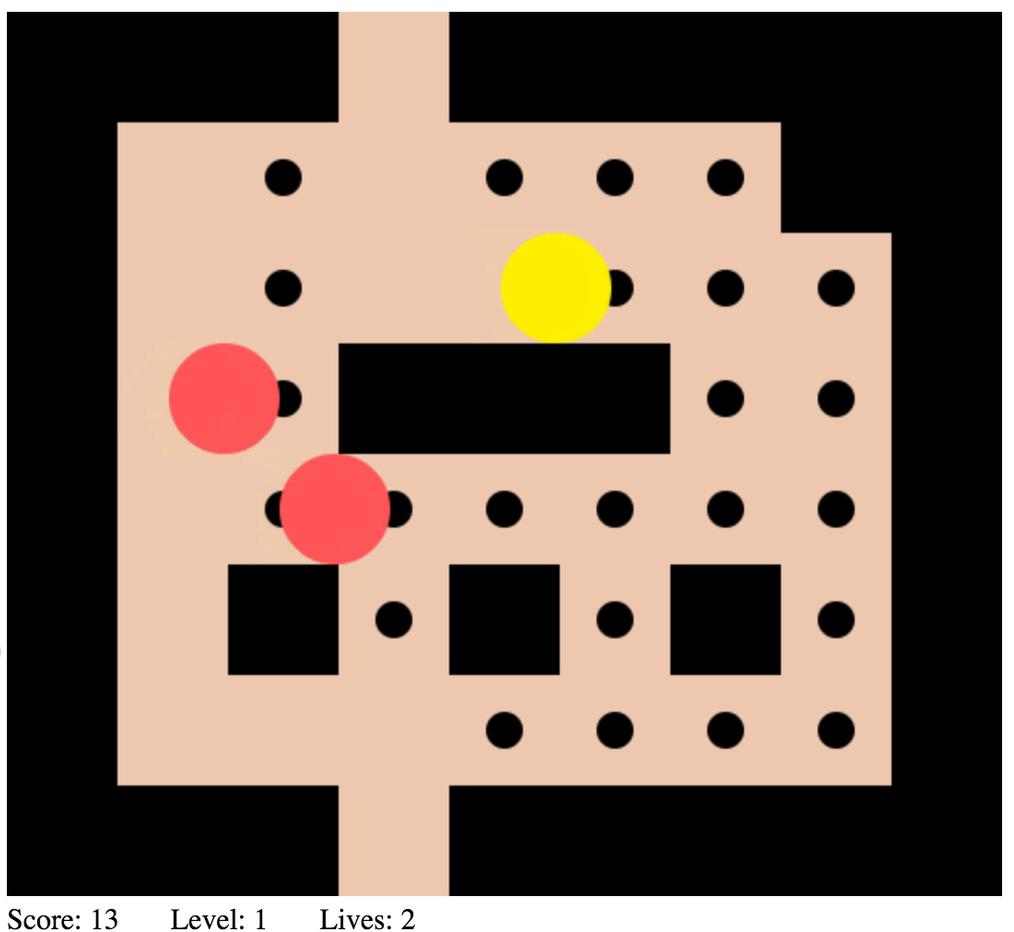
7[1, 1, 1, 2, 1, 1, 1, 1, 1],

8[1, 2, 2, 2, 2, 2, 2, 1, 1],

9[1, 2, 2, 2, 2, 2, 2, 2, 1],

1. [1, 2, 2, 1, 1, 1, 2, 2, 1],
2. [1, 2, 2, 2 ,2, 2, 2, 2, 1],
3. [1, 2, 1, 2, 1, 2, 1, 2, 1],
4. [1, 2, 2, 2, 2, 2, 2, 2, 1],
5. [1, 1, 1, 2, 1, 1, 1, 1, 1],
6. ],
7. startingPac: {
8. X: 1,
9. y: 1
10. },
11. startingGhosts: [{
12. X: 6,
13. y: 6
14. }, {
15. X: 5,
16. y: 1
17. }],
18. telepuerto: **cierto**
19. })

|  |  |
| --- | --- |
| 18: Teleport | 156 |



**Resumen**

En este capítulo se revisa nuestro código de movimiento, se analizaron las estructuras de control dentro de ella, y editado en un momento crucial para permitir el teletransporte. Hemos establecido que para que sólo sería teletransportarse si el nivel especificado que deberíamos, y luego creamos un nivel que se siente más apropiado para el teletransporte.

Antes de dejar este capítulo, quiero tomar un momento para apuntar a algo que no hicimos: No cambiamos nada sobre el algoritmo de movimiento fuera del nextCoordinate. Ese es el poder de abstracción- que acaba de cambiar la forma en que encontramos la siguiente coordenada, y todo lo demás simplemente utiliza los nuevos valores.

En los próximos capítulos vamos a estar girando el mesas- la creación de una bolita de energía que hace que los fantasmas se escapan.

**Una nueva esperanza**

Las semanas se extienden juntos- sueño, correr, comer, correr, colarse, explorar, correr. Los fantasmas son implacables.

Ir a la baja tenía sus ventajas, sin embargo. Los pellets se hizo más densa, lo que permite un mayor gasto energético en la preparación de alimentos. Es decir, más delicioso y alimentos más nutritivos. Cuando usted no estaba mareado por la falta de sueño, usted podría tomar una pequeña cantidad de placer de los sabores deliciosos que no habían estado disponibles en densidades de energía más bajos.

“Los PAC-hombres deben encanta ir hacia abajo tan lejos”, se dice. “Es decir, cuando no hay fantasmas que persiguen después de ellos.”

Jerome asintió con la cabeza. “Esa es una de las ventajas del trabajo. Te dan de comer de las viejas recetas.”‘Las viejas recetas?’

“La densidad pellet, que solía ser como esto todas partes- incluso hasta en la superficie. Es decir, en ese momento la densidad de aquí abajo era probablemente aún mayor. Pero se cosecha los pellets, y cada vez que vuelven a crecer un poco más lento, un poco más pequeño. Es el problema del combustible fósil de nuevo “.

Miras fijamente sin expresión. Tal vez has estado aquí demasiado tiempo para cualquier cosa que tener sentido más. "¿El qué?"

“El problema de los combustibles fósiles. Antes había una energía líquido en la tierra que quemamos para hacer ir las cosas. Al igual que las pastillas, pero aún más denso. Fue maravilloso, o eso dicen las leyendas. Pero seguía cada vez más difícil y más caro para desenterrar, y los efectos de la quema se estaban convirtiendo el planeta ... bueno, ya sabes los santuarios de los árboles? Estaban por todas partes antes. De todas formas, el califato trató de detenerlo, pero finalmente sucumbió a las mismas presiones y deseos que llevaron a los infieles para quemar tanta energía líquida “.

“Y ahora tenemos un mundo desierto donde obtenemos nuestros alimentos de pellets.”

Jerome asintió. "Sí. Pero eso no es absolutamente toda la historia. Todavía podríamos cultivar alimentos sobre el suelo por un tiempo, en lugares en los que no estaban cosechando gránulos. Había más de nosotros entonces “.

No te dan para preguntarle sobre esas últimas declaraciones porque se detecta un fantasma en las proximidades.

Saltar, maniobrar de distancia. Enjuague y repita.

Jerome encuentra otra estación de comunicación, simplemente dice: “Estamos cerca” escuchar cosas Mientras tanto, usted ha comenzado.

“Tenemos un plan”, dicen las voces. Usted no quiere saber lo que este plan es, pero las voces de continuar. “Únase a nosotros, Salam. Hemos elaborado el plan de los primeros principios “.

Los ignora lo mejor que pueda.

Entonces, un día después, finalmente lo encuentra. Un gránulo de poder.

157

|  |  |
| --- | --- |
| Una nueva esperanza | 158 |

“Es más grande que un pellet típica”, dice usted. “Incluso los más densos aquí abajo.” Asiente Jerome. “Gran parte de la energía. No es hermoso?”‘Entonces ... ¿cómo va a encajar?’

“Empezamos a hacer algunas modificaciones para adaptarse a este tamaño de pellet. Hemos reutilizado temporalmente el puerto-térmica de escape cuando se ejecuta en un gránulo de potencia, que va a desactivar la función de escape de calor durante un par de segundos, el tiempo suficiente para obligar en el sedimento. Es kludgy, pero servirá “.

“Así que sólo se extienden sobre el pellet poder?”

"Sí. Haremos un periodo de prueba, y luego se puede llegar a trabajar a experimentar en el siguiente nos encontramos.”Usted se mueve PAC en la parte superior ... y es electrizante.

**19: Potencia**

Así que hemos encontrado las bolas de poder ... ¿Qué hacer con ellos? En este capítulo vamos a colocar las bolas de poder en el mapa, volver nuestra verde Pac basado en, a continuación, hacer frente a un problema que surge de nuestra colocación de pellets de alimentación.

**En el mapa**

En nuestra red actual, utilizamos 0 para representar un cuadrado en blanco, 1 para representar una pared, y 2para representar un pellet. Ahora vamos a añadir3, Que representará un pellet de potencia.

Adición de un pellet de alimentación a un nivel está en fácil como la sustitución de una 2 con un 3- seguir adelante y hacer que en el primer nivel de la matriz de niveles.

Por supuesto, el pellet de potencia no se dibuja o interactuó con todavía, pero eso es lo que nuestros próximos cambios en el código va a hacer.

* *//components/pac-man.js*

2 DibujarRejilla () {

3**dejar** cuadrícula= **esta**.obtener('Level.grid');

4grid.forEach ((fila, rowIndex)=>{

5row.forEach ((célula, columnIndex)=>{

6**Si**(celda== 1) {

7**esta**.drawWall (columnIndex, rowIndex);

* }
* **Si**(celda== 2) {

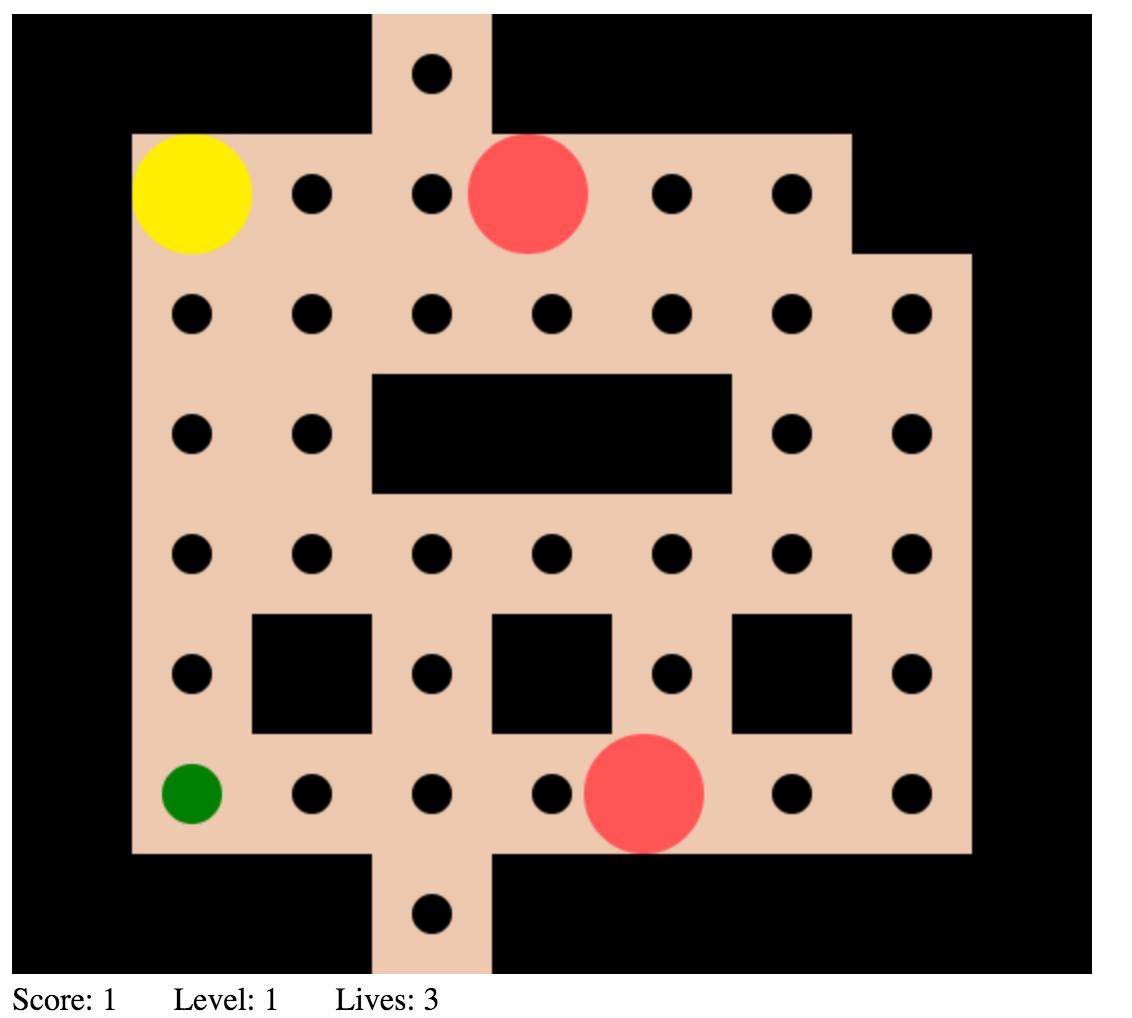
1. **esta**.drawPellet (columnIndex, rowIndex);
2. }
3. **Si**(celda== 3) {
4. **esta**.drawPowerPellet (columnIndex, rowIndex)
5. }
6. })
7. })
8. },
9. drawPowerPellet (x, y) {
10. **dejar** radiusDivisor= 4;
11. **esta**.drawCircle (x, y, radiusDivisor,'detenido','verde')
12. },

159

|  |  |
| --- | --- |
| 19: Potencia | 160 |

Este conjunto de cambios señala a la bolita de energía. EnDibujarRejilla estamos añadiendo un cheque por == celular 3, Que es de la misma forma que los cheques para 1 o 2. Si el cheque coincide, se llama al método nuevodrawPowerPellet, Que es muy similar a la drawPellet pero con un radio diferente y color.

Usted puede preguntar: “¿por qué no usar una pila si / else, en lugar de estos tres IFS separados?” Esa es una gran pregunta, que voy a considerar profundamente al escribir la segunda edición de este libro.



Por lo que la atrae, pero si se mueve el PAC sobre ella, no pasará nada. Tenemos que añadir código que hace que interactúan.

* *//components/pac-man.js*

2 processAnyPellets () {

3**dejar** X= **esta**.obtener('Pac.x');

4**dejar** y= **esta**.obtener('Pac.y');

5**dejar** cuadrícula= **esta**.obtener('Level.grid');

6

7**Si**(GRID [y] [x]== 2) {

8rejilla [y] [x] = 0;

* **esta**.incrementProperty ('Puntuación')

10

1. **Si**(**esta**.obtener('nivel').Esta completo()){
2. **esta**.incrementProperty ('LevelNumber')
3. **esta**.startNewLevel ()
4. }
5. } **else if**(GRID [y] [x] == 3) {

|  |  |
| --- | --- |
| 19: Potencia | 161 |

1. rejilla [y] [x] = 0;
2. **esta**.conjunto('Pac.powerMode', **cierto**)
3. }
4. },

El código que hemos añadido es el último Cláusula la segunda rama en el bloque if / else. Se comprueba para ver si el valor de cuadrícula es 3, y si lo es, se establece el valor a 0. A continuación, establecePowerMode sobre el pac true.

**Modificación de los colores, operadores ternarios**

Que hace PowerMode¿hacer? Bueno, con el tiempo que va a hacer un montón de cosas, pero por ahora vamos a quedar satisfechos con una sola cosa: el cambio de color del Pac.

* *//models/pac.js*

2 PowerMode: **falso**,

3 dibujar(){

4**dejar** X= **esta**.obtener('X');

5**dejar** y= **esta**.obtener('Y');

6**dejar** radiusDivisor= 2;

7**dejar** color= **esta**.obtener('PowerMode')?'# AF0':'# FE0';

8 **esta**.drawCircle (x, y, radiusDivisor, **esta**.obtener('dirección'), color); 9 },

Lo más importante que hemos cambiado en dibujar es la línea siguiente:

* **dejar** color= **esta**.obtener('PowerMode')?'# AF0':'# FE0';

Puede parecer bastante unfamiliar- eso es porque es un operador ternario. Afortunadamente, se puede expresar en términos de una sentencia if / else:

* **dejar** color;
* **Si**(**esta**.obtener('PowerMode')) {

3color = '# AF0';

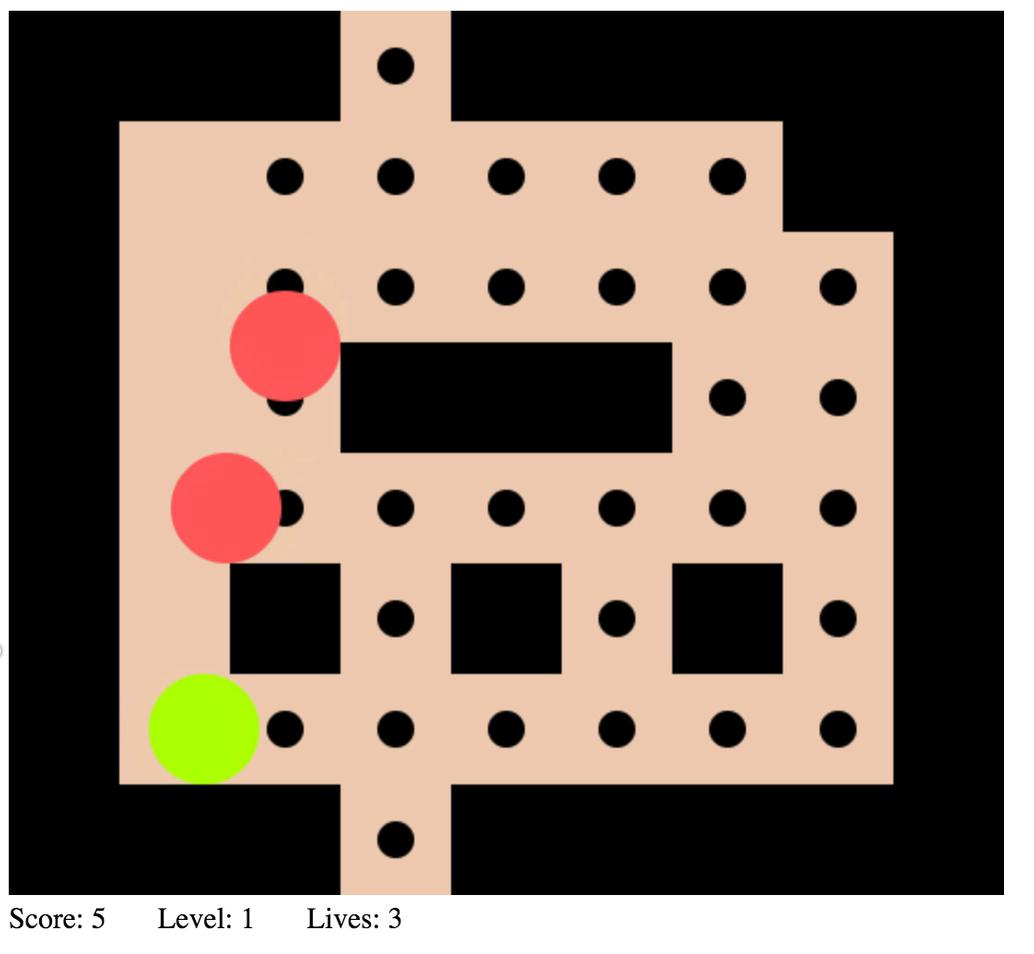
4 } **más** {

5color = '# FE0';

6 }

Como se puede ver, el operador ternario es mucho más corto. También elimina la necesidad de declarar la variable torpemente antes de hacer la instrucción if.

|  |  |
| --- | --- |
| 19: Potencia | 162 |



**El restablecimiento del nivel**

Ahora podemos sacar el sedimento de poder, comemos la pastilla de energía y el cambio de color de la Pac en respuesta, pero hay un error que aparece. Si usted come la pastilla poder, entonces se queda sin vida, se dará cuenta de que la pastilla de encendido no regenerate- que es una pastilla normal ahora.

La razón por la que esto sucede es porque nuestra actual reiniciarfunción simplemente toma cada espacio en blanco y la llena con una bolita de regular. Tiene ningún respeto por la red que nos hemos propuesto.

“Espera un minuto,” usted puede preguntar, “Estamos creando un nuevo nivel cada vez que llamamos startNewLevel, ¿derecho? Y cuando estamos haciendo el nuevo nivel, debe crearlo a partir la red, ¿verdad? Y después de que la rejilla está activado el objeto y no debe afectar a la definición de clase, que se utiliza para crear instancias de los niveles más tarde, ¿verdad?”

Creo que tienes razón en todas esas cosas. Pero el sistema objeto de Ember tiene algunas cuestiones que, cuento largo, significan que cuando se define una matriz como una propiedad de un objeto, diferentes instancias pueden terminar compartiendo los mismos datos. Este es el mismo problema que tuvimos con las pastillas que no aparecen cuando se reinició, es por ello que estamos llamando

level.restart () en startNewLevel.

La forma en que vamos a solucionar este problema es mediante la separación de la red que utiliza el nivel y el diseño de la rejilla. Por lo tanto, como primer paso, cambiar el nombre de todos suscuadrículas en diseños. A continuación, vamos a cambiar la función de reinicio:

|  |  |
| --- | --- |
| 19: Potencia | 163 |

* *//models/level.js*

2 diseño: [

3[2, 2, 2, 1, 1, 1, 1, 1],

* *// ...*

5 ],

6 reiniciar(){

7**var** newGrid=jQuery.extend (**cierto**, [], **esta**.obtener('diseño'));

* **esta**.conjunto('cuadrícula', NewGrid);

9 }

En lugar de bucle a través de cada célula y la sustitución de las cosas, sólo estamos reemplazando la mayor red de una copia de nuestro diseño. Sencillo y eficaz.

El meollo de esto es el uso de jQuery ampliarmétodo. Puede ser utilizado por más, pero aquí estamos usándolo como una forma de copiar el diseño.

El primer argumento de jQuery.extendes un booleano que indica que vamos a hacer una “profunda” se extienden. Esto significa que no sólo va a copiar la capa más superficial (las filas), pero también ir más profundo (en este caso, obtener las células). Eso es importante, porque es las células que queremos!

El segundo argumento es el objeto de destino. Ese es el objeto que va a poner la información copiada en. En este caso se trata de una nueva matriz, lo que garantiza que es un “fresco” javascript constructo que no está relacionada con la disposición (y por tanto cambios en él no tendrá efecto en el diseño).

El tercer argumento es el objeto del que vamos a estar copiando sobre propiedades.

Así tomado todos juntos, estamos copiando sobre diseño en su totalidad a una nueva matriz de JavaScript, a continuación, asignándole a newGrid. cuadrículase ajusta entonces a ese valor. ¡Uf!

**Resumen**

Este nivel fue una mezcla en términos de dificultad. Empezamos con la simple adición de la pastilla de alimentación al mapa, y añadiendo en la interacción. A continuación, pasamos a la introducción de una nueva construcción, el operador ternario, que es básicamente un muy buen atajo para algunos tipos de sentencias if. Finalmente nos dirigimos a un BUG muy espinoso aunque reconoce que no entendemos por qué el error que subyace sucede- y aprendiendo sobre el método de extender jQuery.

En el siguiente capítulo vamos a introducir efectos ofensivos de la bolita de energía. Cuidado con los fantasmas!

**El PAC atacan de nuevo**

Los efectos iniciales de la pastilla de potencia son impresionantes al principio, pero finalmente no tan útil. Se obtiene una descarga de electricidad, y el PAC se vuelve verde.

Afortunadamente, los instrumentos que trajo abajo con que están todavía intactas. Experimentar, primero conseguir las firmas de energía y elementos constitutivos de la pastilla única, a continuación, tratar de combinarlos en algo como arma. Los ancianos tenían razón de que esto tenía potencial. Posiblemente cantidades catastróficas de potencial, pero hoy está centrado en algo para proteger a los PAC de los fantasmas.

Después de un par de horas de medición y el estudio de la pastilla en sí, se inicia la adición de diferentes formulaciones de la pastilla a la PAC para ver cómo cambia las cosas.

Su pellet se agota antes de encontrar algo interesante, pero el más se conoce más numerosas las bolas de poder convertirse. Cada uno lleva a más experimentos, cada uno acercarse a una respuesta, ya sea confirmando una teoría o dejar que se deshaga de uno.

Por último, sucede. Usted salta en el PAC y se puede sentir el poder. Usted escribe la conversión programa- muy rápido, lo suficientemente rápido para ser utilizado en de batalla y cargarlo en la memoria permanente de su PAC.

Jerome envía el programa de conversión de mando. No les hará mucho bien allá arriba, sin bolas de poder, pero será útil si vienen aquí, o si encontrar la manera de preservar bolas de poder el tiempo suficiente para llevarlos hasta allí.

Pero, por ahora, es el momento de la revancha.

164

**20: Cambio de rumbo**

En este capítulo, por fin damos la Pac alguna ofensiva capacidades- la capacidad de ahuyentar a los fantasmas y ganar contra ellos cuando están en PowerMode. Si bien no introducir nuevos grandes conceptos de programación, vamos a ejercer nuestro conocimiento actual y aprender un par de nuevos trozos de sintaxis.

**Santo Run**

Los fantasmas son por lo general te persiguen, pero eso no es necesariamente la mejor medida para ellos después de haber tragado una bolita de energía. Es mucho más prudente para que se ejecuten de distancia.

Haremos esto cambiando el chanceOfPacmanIfInDirection método en el fantasma:

* *// en modelos / ghost.js*
* chanceOfPacmanIfInDirection (dirección) {
* **Si**(**esta**.pathBlockedInDirection (dirección)) {

4**regreso** 0;

5} **más** {

6 **dejar** desirabilityOfDirection = ((**esta**.obtener('Pac.y') - **esta**.obtener('Y')) \* **esta**.GE\ 7 t (`direcciones.**ps**dirección**}**.y`)) +

8 ((**esta**.obtener('Pac.x') - **esta**.obtener('X')) \* **esta**.obtener(`direcciones.**ps**\

* dirección**}**.x`))

1. **Si**(**esta**.obtener('Pac.powerMode')) {
2. desirabilityOfDirection \* = -1;
3. }
4. **regreso** Mates.max (desirabilityOfDirection,0)+ 0,2
5. }
6. },

En primer lugar, hemos cambiado el nombre de lo que solía ser llamado oportunidad a desirabilityOfDirection, Un nombre mucho más descriptivo. Esto se debe a que estamos ya no es sólo calcular si va en esa dirección aumentará las posibilidades de un fantasma de llegar a laPac, Estamos calculando si es una buena idea ir en esa dirección. Y cuando nuestro Pac está en PowerMode la mejor jugada es ejecutar distancia- haciendo la conveniencia de una dirección específica exactamente lo contrario de lo que normalmente sería.

Llevamos a cabo este cambio en el código mediante la comprobación para ver si el Pac está en PowerMode, y luego multiplicando el desirabilityOfDirection variable -1 si esto es.

165

|  |  |
| --- | --- |
| 20: Cambio de rumbo | 166 |

**Los operadores de la taquigrafía**

Aquí estamos viendo una nueva pieza de sintaxis, '\* =':

* **dejar** X= 5
* X \* = 2 *// => 10*

3 X \* = 3 *// => 30*

4 **dejar** y = 5

5 y = y \* 2 *// => 10*

Podemos ver eso '\* ='Es equivalente a multiplicar el valor de la variable del lado izquierdo por el valor del lado de la mano derecha, a continuación, asignar el resultado a sí mismo. Es un operador de taquigrafía.

operadores de taquigrafía similares se pueden encontrar para otros operadores de Javascript.

* **dejar** X= 5
* X + = 2 *// => 7*

3 X - = 3 *// => 4*

4 X / = 2 *// => 2*

operadores de taquigrafía pueden hacer que el código tanto más fácil de escribir y más fácil de leer.

**Las colisiones más específicos**

Los fantasmas se están ejecutando, y que están haciendo eso porque saben que si colisionan con el Pac mientras está en PowerMode, van a tener que retirarse por completo del campo de batalla.

Con el fin de representar a esas consecuencias en el juego, vamos primero tenemos que cambiar nuestra collidedWithGhost Método de manera que en lugar de devolver solo cierto o falso, Devuelve detalles sobre la cual fantasma chocó con nuestra Pac. Antes no importaba, ya que las únicas consecuencias fueron para el Pac, pero ahora los propios fantasmas tener consecuencias.

También vamos a renombramos detectGhostCollisions, ya que collidedWithGhost implica que la función devuelve una respuesta verdadera / falsa.

|  |  |
| --- | --- |
| 20: Cambio de rumbo | 167 |

* detectGhostCollisions () {
* **devolver este**.obtener('fantasmas') .Filter ((fantasma)=>{
* **regreso** (**esta**.obtener('Pac.x')==ghost.get ('X')&&

4 **esta**.obtener('Pac.y')==ghost.get ('Y'))

* })

6 },

Como se puede ver si se compara con el código actual, es casi exactamente lo mismo pero con una gran diferencia: en lugar de utilizar el alguna método, estamos usando la filtrarmétodo. Así que en lugar de devolver un valor lógico, nos devuelve una matriz de los elementos cuya función de filtro vuelto realidad.

Si uno Ghost coincidía con la función que alimenta a filtrar, Entonces la matriz devuelta tendría que un fantasma en ella. Si dos fantasmas emparejados, la matriz devuelta tendría dos fantasmas en ella. Muy a menudo se devolverá una matriz vacía, ya que la mayoría de las veces no estamos corriendo en cualquier fantasmas.

Cuando estamos usando detectGhostCollisions dentro del bucle, tendremos que cambiar nuestra técnica un poco cuando estamos comprobando para ver si existen colisiones.

* lazo(){
* *// ...*

3

4**dejar** ghostCollisions= **esta**.detectGhostCollisions ();

5**Si**(ghostCollisions.length> 0) {

* **Si**(**esta**.obtener('Pac.powerMode')) {
* ghostCollisions.forEach (fantasma => ghost.retreat ())

8} **más** {

* **esta**.decrementProperty ('vive');

1. **esta**.reiniciar();
2. }
3. }

13

1. Ember.run.later (**esta**, **esta**.lazo, 1000-1060);
2. },

Ahora en vez de pasar en el booleano de la instrucción if, estamos comprobando la longitud y ver si está por encima de cero. También asignamos a una variable para que podamos volver a utilizar la gama más adelante sin correrdetectGhostCollisions dos veces.

Después de haber determinado si hay colisiones de fantasmas que necesitamos para comprobar si el Pac está en PowerMode. Si no es así, vamos a hacer lo mismo que hemos vidas decremento siempre hecho-, a continuación, reinicie todo. Si es en PowerMode, vamos a recorrer todos los fantasmas chocaron y contarles todo a retirarse.

|  |  |
| --- | --- |
| 20: Cambio de rumbo | 168 |

**Retiro completa**

Cuando llegamos a un fantasma, queremos retirado del campo de juego que Santo. Que la eliminación se hará temporal en el siguiente capítulo (y muchos de estos detalles cambiará), pero en este capítulo sólo estamos diferenciando entre eliminado y el presente.

Vamos a empezar por dar a nuestros fantasmas en un booleano remoto y cambiando la forma en que actúa a partir de la situación de que booleano

* *//models/ghost.js*

2 remoto: **falso**,

3 retirada(){

* **esta**.conjunto('remoto', **cierto**)

5**esta**.conjunto('X',-1)

6**esta**.conjunto('Y',-1)

7 },

8

9 *//mixins/movement.js*

1. movimiento(){
2. **Si**(**esta**.obtener('remoto')) {
3. *// no hacer nada, se ha ido*
4. } **else if**(**esta**.animationCompleted ()) {
5. *// ...*
6. }

remoto es false por defecto (también esta Falsy por defecto en todas partes, pero esto hace que sea explícita). Entonces cuando llamamos retirada en un fantasma, se establece remoto a cierto. Esto tiene el efecto de hacer que el fantasma eliminado inmóvil, logrado mediante la creación de una rama nula en la parte superior de un gran si / cadena demás enmovimiento.

También en el retirada la función que estamos estableciendo la posición del fantasma [-1, -1]... con seguridad fuera del tablero, por lo que el fantasma no se quede atascado en una plaza del Pac necesita.

**Reagruparse**

En la versión comercial de Pac-Man, los fantasmas no desaparecen cuando eaten- van a un lugar específico en el tablero a la espera de su oportunidad de volver a salir. Vamos a repetir aquí que mediante la especificación de esos lugares en el documento de nivel, entonces el uso que en elretirada función.

|  |  |
| --- | --- |
| 20: Cambio de rumbo | 169 |

* *// en modelos / teletransporte-level.js*

2 **importar** nivel de './nivel';

3

4 **por defecto de exportación** Level.extend ({

* diseño: [
* [1, 1, 1, 2, 1, 1, 1, 1, 1],

7[1, 2, 2, 2, 2, 2, 2, 1, 1],

8[1, 2, 2, 2, 2, 2, 2, 2, 1],

9[1, 2, 2, 1, 1, 1, 2, 2, 1],

1. [1, 2, 2, 2 ,2, 2, 2, 2, 1],
2. [1, 2, 1, 2, 1, 2, 1, 2, 1],
3. [1, 3, 2, 2, 2, 2, 2, 2, 1],
4. [1, 1, 1, 2, 1, 1, 1, 1, 1],
5. ],
6. *// ...*
7. ghostRetreat: {
8. X: 4,
9. y: 3
10. }
11. })

En primer lugar se define el ghostRetreathachís, lo que le da una X y un valor de y. Esta coordenada es donde vamos a poner en retirada a los fantasmas. Tenga en cuenta que en este nivel, está en medio de una pared- en alguna parte que el Pac no tendrá que ir con el fin de completar el nivel. Esa pared es también justo al lado de un espacio abierto, lo cual es importante para cuando el fantasma finalmente decide salir de su retiro (capítulo siguiente).

Ahora vamos a utilizar el ghostRetreat al retirarse.

* *//models/ghost.js*

2 retirada(){

* **esta**.conjunto('remoto', **cierto**)

4**esta**.conjunto('FrameCycle',0)

5**esta**.conjunto('X', **esta**.obtener('Level.ghostRetreat.x'))

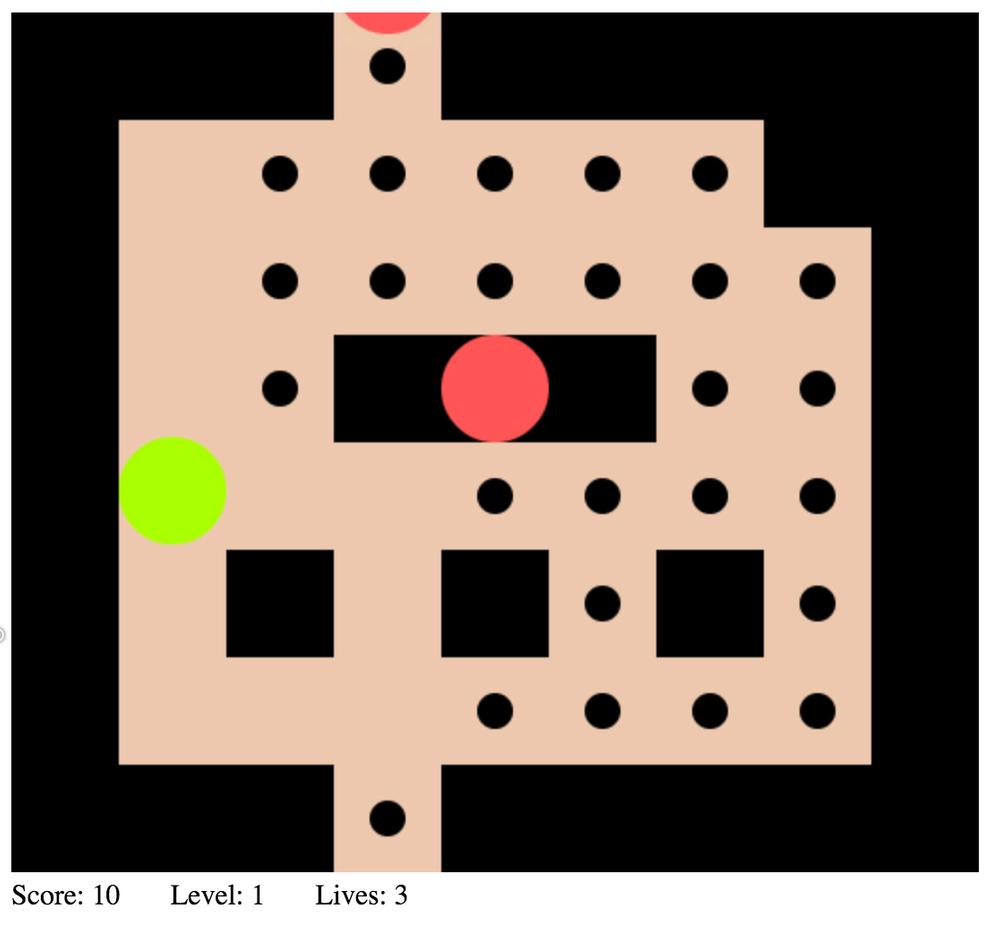
6**esta**.conjunto('Y', **esta**.obtener('Level.ghostRetreat.y'))

7 },

Estamos estableciendo el X y y valor a los valores de la ghostRetreatpicadillo. Eso va a poner en el designadoghostRetreat cuadrado.

También estamos estableciendo el frameCycle a 0- lo contrario será como si el fantasma es en animación suspendida a mitad de camino de la plaza de retiro (probarlo para ver el efecto).

|  |  |
| --- | --- |
| 20: Cambio de rumbo | 170 |



Asegúrese de especificar una ghostRetreat cuadrado en cualquier nivel que tiene fantasmas y una bolita de energía.

**Resumen**

En este nivel hemos ganado el poder de la bolita de energía ... pero ninguno de sus limitaciones. Esas limitaciones se pondrá al día con nosotros el próximo capítulo cuando se descubre que la bolita de energía no puede durar para siempre.

Pero mientras tanto, podemos tomar el sol en nuestra invencibilidad temporal Pac, nuestra habilidad con la mejora de programas complejos, y nuestro conocimiento de la nueva sintaxis \* = y filtrar.

**El retorno del jihadista**

La persecución continúa, rompiendo uno tras otro fantasma. Intenta dar vuelta a sus ojos lejos de los restos de tentáculos cada máquina, pero se pega fragmentos de imágenes en su mente. A veces ves las piernas hacia afuera. Brazos. Es difícil de destruir el fantasma sin matar a todo lo que hay dentro.

A veces, se puede oír gritando.

“Nos perdonamos”, dicen las voces. "Ven a nosotros."

Demasiado tarde se ve lo que debería haber sido obvio- que un pellet de potencia individual no puede durar para siempre. Usted barril en un fantasma, pero esta vez que no se tuerce y se doble bajo su ataque. Los tentáculos se recogen y retienen el PAC todavía, mientras que otros abusen de capa tras capa de la PAC.

“Salam”, dice Jerome, “Lo siento. Siento no haber podido proteger. Por favor cree. Antes de que sea demasiado tarde. Quiero verte en el cielo. Quiero que cosechar los frutos de nuestra Yihad “.

La radio se corta y ya está al aire libre, agarró y se abrió por uno de los tentáculos. “Cooperar”, dice una voz. “Cooperar, y vamos a dejar que su amigo ir.”

Miras a tu alrededor, al menos tanto como le sea posible mientras colgando en el aire. PAC es destruida, junto con los instrumentos científicos. Jerome es en caja, pero aún haciendo carreras suicidas en los fantasmas que bloquean su camino. Lo rechazo, pero que fácilmente podían destruir.

No hay nada que decir, pero 'sí'.

171

**21: Se acabó el tiempo**

Las cosas buenas no pueden durar para siempre. Esto es válido tanto para este libro y por la invencibilidad PowerMode de nuestra Pac.

No se preocupe, vamos a terminar la invencibilidad antes de terminar el libro.

**temporizadores**

Este capítulo sobre todo consistir en varias discusiones bastante complejas, mientras que establecimos temporizador PowerMode del Pac y los indicadores visuales que ayudan a nuestros usuarios a resolverlo. Luego, al final vamos a eco de esas discusiones para temporizador de retroceso de nuestro Santo.

En primer lugar, vamos a repasar la arquitectura temporizador general, lo que vamos a poner en nuestra movimiento mixin.

* *//mixins/movement.js*

2 temporizadores: [],

3 movimiento(){

* *// ...*

5

* **esta**.tickTimers ();7 },

8 tickTimers () {

9**esta**.obtener('' temporizadores) .forEach ((timerName)=>{

1. **Si**(**esta**.get (timerName)> 0) {
2. **esta**.decrementProperty (timerName)
3. }
4. })
5. },

Tenemos una gran variedad de timers- vacío por defecto, pero vamos a llenarlos lo suficientemente pronto. Luego, al final de lamovimiento función (llamada una vez cada vez a través del bucle) que llamamos tickTimers.

tickTimers pasa por cada contador de tiempo y, si es mayor que 1, disminuye la propiedad en 1. Porque obtener y decrementPropertyambos toman cuerdas, el mismo bucle puede disminuir cualquier propiedad arbitraria cuyo nombre que le demos, que es una forma increíblemente eficiente de hacer las cosas. De esta manera, el mismotickTimers función puede trabajar tanto para los temporizadores y los contadores de tiempo Pac Ghost.

Así que vamos a crear un temporizador para nuestra Pac- éste cuenta atrás cuánto tiempo estaremos en PowerMode.

172

|  |  |
| --- | --- |
| 21: Se acabó el tiempo | 173 |

**el poder menguante**

En primer lugar vamos a definir algunas variables.

* PowerMode: Ember.computed.gt ('PowerModeTime', 0),

2 powerModeTime: 0,

3 maxPowerModeTime: 400,

4 temporizadores: ['PowerModeTime'],

Estamos estableciendo cuatro propiedades aquí. La primera está redefiniendoPowerMode, Que ahora en lugar de ser sólo una cierto o falso valor, que es una propiedad que devuelve un computarizada cierto o falsovalor que depende de varios factores. En este caso, si elpowerModeTime propiedad es mayor que ( 'gt' para abreviar) 0, entonces

PowerMode es cierto.

¿Quieres saber más acerca de las propiedades calculadas? [Haga clic aquí para ver mis videos en](https://www.emberscreencasts.com/tags/computed-properties) [them¹³](https://www.emberscreencasts.com/tags/computed-properties).

La segunda propiedad es que estamos estableciendo powerModeTime, Que es la cantidad de tiempo que nos queda en PowerMode. Se inicia en 0 y, como veremos en breve, a mayores exigencias cuando recogemos una bolita de energía.

La tercera propiedad es maxPowerModeTime, Que en este caso es 400. Dado que estamos recibiendo aproximadamente 60 garrapatas a través del bucle por segundo, esto nos duran de 6 a 7 segundos.

La última propiedad es la matriz temporizadores. El valor por defecto demovimiento es una matriz en blanco, pero aquí estamos añadiendo powerModeTime. Eso es diciéndole que se marque esa propiedad por uno cada vez a través del bucle durante nuestratickTimers llamada.

Hay otra importante pieza de código que se ejecuta cada vez que comemos la pastilla de potencia:

* *// en componentes / pac-man.js*
* *// carrera al comer la pastilla de potencia, en sustitución de `this.set ( 'pac.powerMode', true)`*

3 **esta**.conjunto('Pac.powerModeTime', **esta**.obtener('Pac.maxPowerModeTime'));

Aquí es donde empieza todo. En vez de girarPowerMode true directamente, lo estamos haciendo de forma indirecta mediante el establecimiento powerModeTime a un valor mayor que 0.

Si se ejecuta el juego ahora, usted debe ser capaz de comer una pastilla de alimentación, espere un par de segundos, y ser normal de nuevo. Sin embargo, eso puede ser frustrante para el jugador (y bastante devastador para nuestro personaje principal) cuando están persiguiendo un fantasma y la pluma, que son normales otra vez y mueren.

Vamos a añadir algunos indicadores de que el jugador puede ver.

**Un nuevo caso**

El primer indicador será el más fácil.

¹³<https://www.emberscreencasts.com/tags/computed-properties>

|  |  |
| --- | --- |
| 21: Se acabó el tiempo | 174 |

* <**lona** carné de identidad="myCanvas" anchura={{}} Level.pixelWidth altura={{}} Level.pixelHeight></ \

2 canvas>

3 <**br**>

4 Puntuación: {{score}} **& Nbsp; & Nbsp; & Nbsp;** Nivel: {{}} levelNumber **& Nbsp; & Nbsp;** ynbsp \

5 ; Vidas: {{}} vidas **& Nbsp; & Nbsp; & Nbsp;**

6 {{#if pac.powerMode}}

7PowerMode: {{}} pac.powerModeTime

8 {{/Si}}

Aquí hemos añadido algo de código para el final de nuestro archivo de manillar que muestra el powerModeTime variable sobre el Pac ... pero sólo lo hace si pac.powerMode es verdad.

La construcción que estamos utilizando para hacer eso es un manillares si el bloque. El hash (#) Desde el principio nos dice que se trata de un bloque de manillar, por lo que la {{}} Argumento #ifsección es lo que nos dice si se trata de un bloque. El bloque si termina en{{/Si}}. Nada en el medio es el bloque, y se muestran si y sólo si el argumento dado en la declaración bloque si es cierto.

Es un poco como nuestro Javascript sentencia if, pero en forma manillar.

**Modificación de los colores**

El segundo indicador será un poco más difícil, e implicará el aprendizaje de varias nuevas construcciones.

Lo que vamos a hacer es tener el Pac desvanecen lentamente de color verde brillante a amarillo, basado en la cantidad de tiempo que queda en el temporizador. No es que mucho código, pero está lleno de cosas nuevas avanzada, así que voy a volcar aquí y luego ir sobre ella el concepto por concepto.

* *//models/pac.js*

2 dibujar(){

3**dejar** X= **esta**.obtener('X');

4**dejar** y= **esta**.obtener('Y');

5**dejar** radiusDivisor= 2;

6 **esta**.drawCircle (x, y, radiusDivisor, **esta**.obtener('dirección'), **esta**.obtener('color')); 7 },

8 color: Ember.computed ('PowerModeTime', **función**() {

9**dejar** timerPercentage= **esta**.obtener('PowerModeTime')/ **esta**.obtener('MaxPowerModeTime');

1. **dejar** motorizado={r: 60, g: 100, b: 0};
2. **dejar** normal={r: 100, g: 95, b: 0};
3. **dejar** [R, g, b]=['R','gramo','segundo'].mapa(**función**(RgbSelector) {
4. **dejar** color=accionada [rgbSelector]\*timerPercentage+

14 normal [rgbSelector] \* (1 - timerPercentage)

1. **regreso** Mates.round (color)
2. })

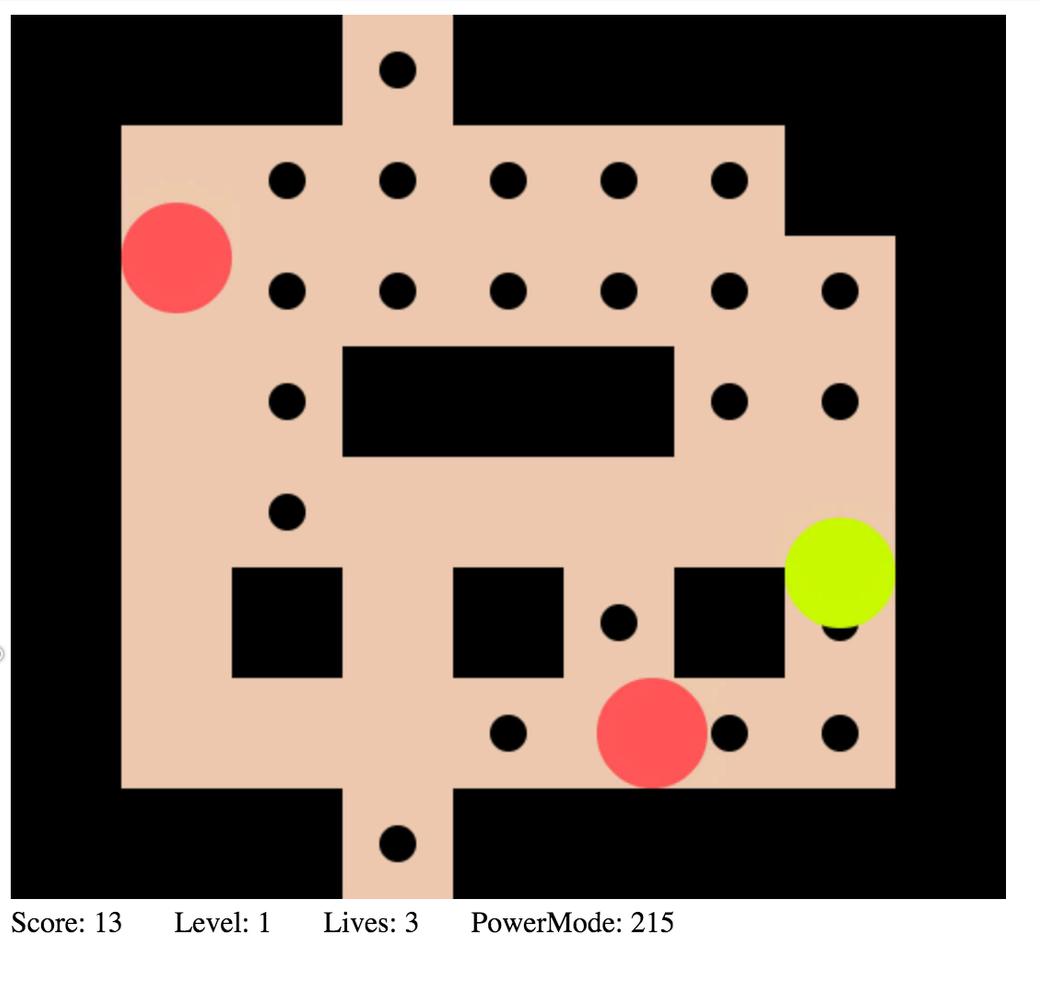
|  |  |
| --- | --- |
| 21: Se acabó el tiempo | 175 |

1. **regreso** `Rgb (**ps**r**}**%,**ps**gramo**}**%,**ps**segundo**}**%) `
2. }),

El único cambio en dibujar es el hecho de que cada vez es su color de la propiedad computarizada color en lugar de definir con un operador ternario (de fantasía sentencia if) como estábamos antes.

Entonces tenemos la propiedad computarizada color. Tiene una clave dependiente delpowerModeTime, Lo que significa que si la propiedad powerModeTime ha cambiado desde la última vez colorpropiedad ha sido llamado, entonces se vuelve a calcular. De lo contrario, puede utilizar el mismo valor que antes.

Los grandes rasgos de esto son que tenemos una timerPercentage, Un número entre 0 y 1, que corresponde a la cantidad de la energía que nos queda. Entonces tenemos dos hashes (motorizado y normal) Que especifican la cantidad constituyente del rojo, verde y azul que se requiere para hacer la colores con alimentación y normales respectivamente (verde y amarillo brillante). A continuación, la próxima próxima gran bloque se trata de encontrar las cantidades actuales de rojo, verde y azul para mezclar, sobre la base de mezcla de los colores normales con alimentación y en un porcentaje determinado en base a la cantidad de energía que queda. Finalmente lo ponemos en una forma que pueda entender CSS.



No se preocupe si usted no recibió todos los que- era sólo un resumen, y vamos a ir a través de varios de los conceptos con más detalle.

**RGB**

En nuestras relaciones anteriores habíamos utilizado códigos hexadecimales para nuestros colores. Pueden ser códigos de 6 dígitos compuestos por los números 0-9 y AF, pero que habían estado utilizando los códigos de 3 dígitos más simplistas, como# EF0 y #FAA. El símbolo de hash, en este contexto, indica que estamos usando hexadecimal.

|  |  |
| --- | --- |
| 21: Se acabó el tiempo | 176 |

Para nuestra propiedad calculada color, vamos a cambiar a porcentajes RGB. Son un poco más detallado, pero son mucho más fáciles de razonar acerca y combinar.

Rápido, ¿cuál es el promedio de 40 y 60? Bien, ahora ¿cuál es el promedio de 9 y C? El primero de ellos es mucho más fácil, porque los seres humanos no piensan de forma natural en hexadecimal. (Las respuestas son '50' y 'A8')

En cuanto a estos dos valores hash, podemos tener una pista decente en lo que van a ver como y cuando estén listas:

* **dejar** motorizado={r: 0, g: 100, b: 0};

2 **dejar** normal = {r: 100, g: 95, b: 0};

Podemos ver aquí a simple vista que el estado de alimentación será muy verde, y el estado normal será una mezcla de rojo y verde (que termina siendo de color amarillo). La parte media después toma un promedio ponderado (más sobre esto en breve) y se alimenta a que la sintaxis de color RGB:

* **regreso** `Rgb (**ps**r**}**%,**ps**gramo**}**%,**ps**segundo**}**%) `

A completamente “normal” PAC (uno con una powerModeTime de 0) devolvería rgb (100%, 95%, 0%).

**la mezcla de colores**

La parte central es donde la magia happnes- tomamos un promedio ponderado basado en la cantidad de energía que hay. Eso es lo que la siguiente parte se trata.

* **dejar** [R, g, b]=['R','gramo','segundo'].mapa(**función**(RgbSelector) {

2**dejar** color=accionada [rgbSelector]\*timerPercentage+

3 normal [rgbSelector] \* (1 - timerPercentage)

* **regreso** Mates.round (color)5 })

Hay dos nuevas piezas aquí, así como algunos algoritmos no-super-intuitivas pasando. Vamos a tomar las cosas una a la vez.

La primera parte es nueva Math.round, Pero es bastante explica por sí mismo.

La segunda parte nueva es array desestructuración. Aquí hay un ejemplo muy sencillo.

|  |  |
| --- | --- |
| 21: Se acabó el tiempo | 177 |

* **dejar** [R, g, b]=['rojo','gorila','Boxer'];

2 console.log (g)*// => 'gorila'*

Básicamente, estamos asignando valores de varias variables en función de su posición en la matriz receptora. En nuestro ejemplo más complicado, estamos asignando esas variables sobre la base de lo que está de regresar de la función dentro de lamapa.

Esa función es donde se obtienen los promedios ponderados individuales. losrgbSelectores o bien 'r', 'g', o 'b', y se puede encontrar el número correcto de la motorizado y normalhashes. Entonces están multiplicados por la cantidad de energía o bien se deja (para elmotorizado hash) o la cantidad de energía que se haya drenado (para el normalpicadillo). Los que se suman. El efecto entero es de la misma transición de un color a otro como el temporizador hace una cuenta regresiva a 0.

**Retirarse, Redone**

Podemos tomar toda la lógica que acabamos acostumbrados a plazo y un código de colores PowerMode del Pac y utilizarlo para plazo y código de color de la retirada de los fantasmas.

En primer lugar vamos a añadir en varias propiedades en el Santo;

* *//models/ghost.js*
* remoto: Ember.computed.gt ('RetreatTime', 0),

3 retreatTime: 0,

4 maxRetreatTime: 500,

5 temporizadores: ['RetreatTime'],

Esas podrían resultar familiar, porque son básicamente el mismo conjunto de propiedades como PowerMode, Sólo que con diferentes nombres. De la misma manera, vamos a cambiar una línea enretirada de establecer la propiedad en true para ajustar el tiempo al máximo, al igual que cambiamos la línea al recoger un pellet de energía.

* *//models/ghost.js*

2 retirada(){

* **esta**.conjunto('RetreatTime', **esta**.obtener('MaxRetreatTime'))
* *// ...*

5 }

los retreatTime a continuación, la variable está marcada a 0 por la tickTimers método, que se ejecuta automáticamente en cada bucle para cada temporizador dentro de temporizadores.

El color de la Pac se establece a continuación, de una manera similar:

|  |  |
| --- | --- |
| 21: Se acabó el tiempo | 178 |

* *//ghost.js*
* color: Ember.computed ('RetreatTime', **función**() {
* **dejar** timerPercentage= **esta**.obtener('RetreatTime')/ **esta**.obtener('MaxRetreatTime');

4**dejar** retirado={r: 0, g: 0, b: 0};

5**dejar** normal={r: 100, g: 40, b: 40};

6**dejar** [R, g, b]=['R','gramo','segundo'].mapa(**función**(RgbSelector) {

7**dejar** color=retirado [rgbSelector]\*timerPercentage+

8 normal [rgbSelector] \* (1 - timerPercentage);

* **regreso** Mates.round (color);

10});

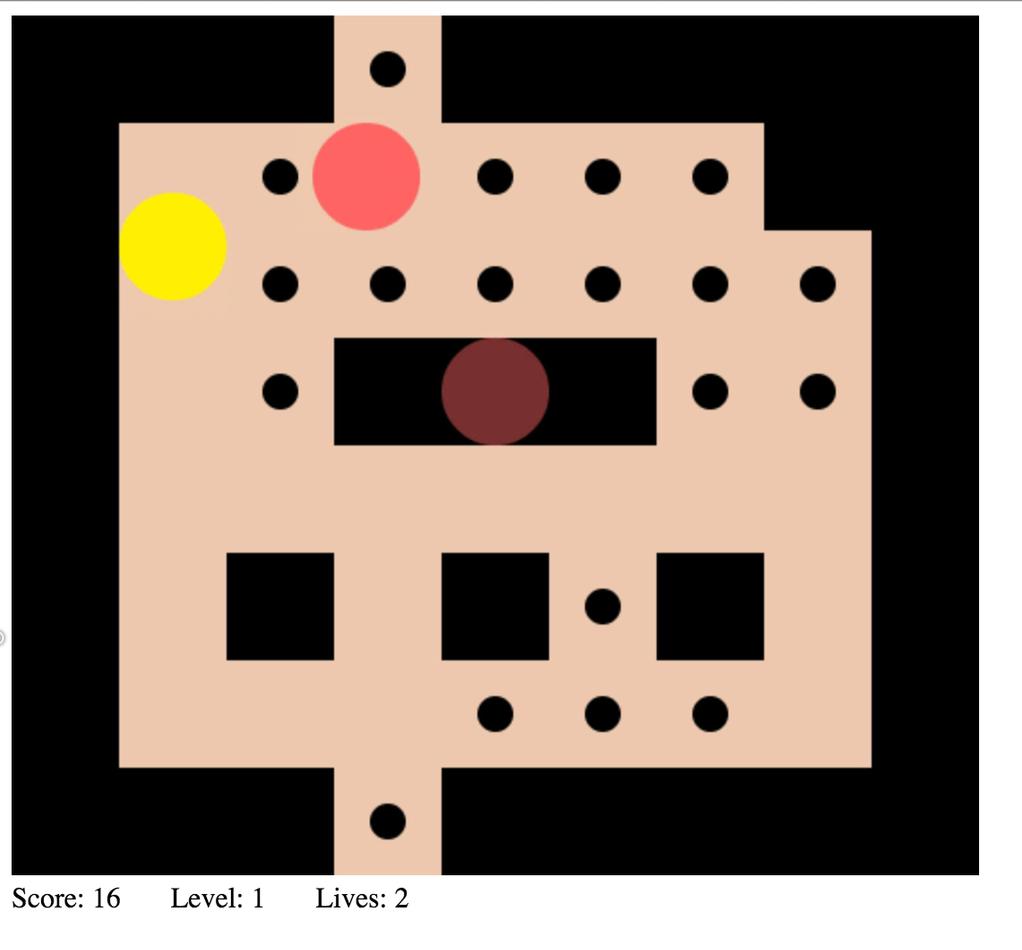
11 **regreso** `Rgb (**ps**r**}**%,**ps**gramo**}**%,**ps**segundo**}**%) `; 12 }),

Los números han cambiado, pero la mayoría de las otras cosas sigue siendo el mismo.

asignación de bonificación: resumen tanto de los dos color métodos como sea posible para un método común de que sólo se necesita en los argumentos de los dos extremos de color y la

timerPercentage.

Ahora, cuando los fantasmas se retiran, se dirigen a la plaza de retiro, a su vez negro, y poco a poco se desvanecen de nuevo a rojo. Es más bien un efecto nefasto. Entonces, cuando están totalmente rojo otra vez, se ponen de nuevo en el juego.



**Resumen**

Hemos terminado con el juego! ¡Hurra!

|  |  |
| --- | --- |
| 21: Se acabó el tiempo | 179 |

Este último capítulo era una grande, que cubre una gran cantidad de nuevos temas como el esquema de codificación de color RGB, matrices desestructuradas, y el manillar bloquear ayudantes. Hemos añadido en temporizadores para cambiar el estado de las PAC y los fantasmas, dándoles efectos que se desgastan lentamente.

A pesar de que este es el último capítulo, todavía hay más que se pueda hacer. Voy a hacer un complemento de lo que puede incluir fácilmente en su aplicación. Voy a incluir en ese complemento de un conjunto de niveles que se sienten más como los niveles tradicionales Pacman.

Luego, por supuesto, hay mucho más que se podía hacer para modificar el modo de juego. ¿Quién ha dicho PacMan fue la mejor versión posible de PacMan? Tal vez puede que sea más divertido! Si usted piensa que puede, ahora tiene las herramientas para hacerlo.

¿Ha mejorar PacMan? Envíame tu mejor tiro. Quiero hacer una lista de variantes PacMan fresco creado por los lectores.

**Epílogo**

“¿Por qué nos matan?” Le pregunte.

Los hmm fantasmas y haw, moviéndose incómodo. Ambos están mirando fijamente a la pantalla del ordenador. “Un momento,” dice el llamado Elos. “Estamos tratando de poner esto en palabras que su cultura puede entender.”

El segundo, Damian, asiente con la cabeza por delante de su pantalla. “Éste es extraño por su cultura, más parecidos a nosotros. Es por eso que le hemos seleccionado, después de todo. Él puede ser capaz de entender.”Damian dice esto con apenas un cambio en la expresión.

“Muy bien”, dice Elos. “El planeta ha sido destruido, al igual que el profeta temía.” “El profeta ha trabajado a cabo a partir de primeros principios.”

“Por lo tanto,” usted dice, “Eso no dice por qué has estado matando a nosotros.”

“El profeta tenía un sueño”, dice Elos. “Un sueño de vida sostenible de la luz solar, la humanidad se viaja entre las estrellas.”

“Un sueño muy extraño en el momento, pero el libro sagrado dice que trabajó a cabo a partir de primeros principios.”

Ambos se ven desde sus pantallas, te quedan mirando, pues, dicen al unísono: “Se ha cumplido, aunque sólo en parte.”

Miran hacia atrás hacia abajo en sus pantallas.

“Su civilización no se ha iniciado esta destrucción- planetario que fue la civilización antes de la una en la que nuestro profeta fue nacido de la civilización, pero su casi se ha terminado.”

“Y eso es una excusa para matarnos a todos?”

“Dentro de tres generaciones, que le todos muertos de todos modos. Los gránulos se han secado, la tierra completamente estéril. Matando a algunos de ustedes ahora era necesario, para que después muchos pueden vivir.”

“Es por eso que envió el diluvio de ... de lo que se llama a los fantasmas.”

“Así que usted mata a la mitad de nosotros antes de que nos matamos a nosotros mismos?” Le pregunte. "¿Es asi?"

“Oh, no, eso no es ciertamente toda ella,” dice Elos y sin expresión. “Hemos matado a más de la mitad. Y no hemos terminado “.

“Sí”, dice Damian. “Tenemos un plan más grande. Un plan para restaurar la tierra a su antigua gloria: o mejor dicho, a una gloria mucho mayor de lo que ha visto en su vida. Una sociedad donde los hombres son pacíficos y libres “.

“No te gustará este plan, no al principio. Va a ser reticentes a ayudarnos, a pesar de su ayuda dará lugar a mucho menos ... colateral. Creo que de esta manera- estamos separando las ovejas de las cabras “.

“Y este plan”, añade Damian, “Hemos trabajado a cabo a partir de primeros principios.”

180

**¿Que sigue?**

Hemos construido un gran juego de PacMan ... ¿y ahora qué?

En primer lugar, si desea continuar sus aventuras PacMan, echa un vistazo [la addon¹⁴ PacMan](https://github.com/jeffreybiles/pac-man) (Derivado del código de este libro) y [mi screencast sobre cómo personalizar los niveles en el addon¹⁵ PacMan](https://www.emberscreencasts.com/posts/132-customizing-the-pacman-addon) (Que ha cambiado un poquito, para que los niveles de desarrollo más grandes más fácil, y el conjunto de nivel por defecto es similar al del original, Ms. Pac-Man). El uso de estos, puede crear fácilmente sus propios niveles y compartirlos con el mundo.

Si suficiente gente compra este libro, puedo escribir una sequel- que contará la historia de los tiempos de Salam con los fantasmas, y mostrar cómo crear un editor de niveles PacMan usando Ember. Así que si te ha gustado este libro, decirle a sus amigos!

A continuación, si usted está queriendo saber más sobre Ember, hay un montón de grandes recursos para que usted compruebe hacia fuera.

* [EmberScreencasts.com¹⁶](https://www.emberscreencasts.com/). Hay más de 100 vídeos cortos que enseñan diversos aspectos de Ember y tecnologías (incluyendo algunos ES6) relacionados, y libero a dos nuevos videos cada semana.
* [El Ember Guides¹⁷](https://guides.emberjs.com). Las guías oficiales. Bien escrito, y bastante bueno para comprobar cuál es el de manera oficial es hacer las cosas.
* [Ember 101¹⁸](https://leanpub.com/ember-cli-101). Un libro electrónico gratuito que se centra en EmberJS.

Y, por supuesto, hay mucho más esperando a ser descubiertos, y cosas nuevas sucediendo todos los días. yo recomiendo[Ember Weekly¹⁹](http://emberweekly.com/) para mantenerse al día con la última.

¹⁴<https://github.com/jeffreybiles/pac-man>

¹⁵<https://www.emberscreencasts.com/posts/132-customizing-the-pacman-addon>

¹⁶<https://www.emberscreencasts.com/>

¹⁷<https://guides.emberjs.com>

¹⁸<https://leanpub.com/ember-cli-101>

¹⁹<http://emberweekly.com/>

181