
ORACLE
Academy

Java Foundations

2-1

El Proceso de Desarrollo de Software

ORACLE Academy

Objetivos

- En esta lección se abordan los siguientes objetivos:
  - Comprender el modelo espiral de desarrollo
  - Reconocer las tareas y subtareas del modelo espiral
  - Saber qué ocurre cuando se ignoran pasos
  - Identificar las funciones de software
  - Comprender cómo se implantan gradualmente las funciones

Ejercicio 1, Parte 1

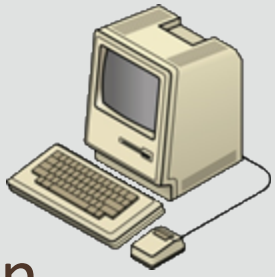
- Su amigo Carlos tiene planes para el fin de semana
- Eche un vistazo a su mensaje de correo y decida qué pasos serían necesarios tomar para que puedan llevarse a cabo estos planes:

Hola:

Al parecer, este mes hay una exposición sobre la historia de la informática en el museo municipal, y algunos del grupo estamos pensado en ir el viernes a las 17:00. ¿Te apuntas? Creo que el metro sería la forma más rápida de llegar

Carlos

Ejercicio 1, Parte 2



- Complete la tabla; debe escribir como mínimo un elemento en cada sección

Requisitos

- ¿Qué pregunta Carlos en su mensaje de correo?

Diseñar un plan

- ¿Qué cosas debe tener en cuenta antes de salir?

Pruebas

- ¿Cómo sabe que el plan funcionó?

Ejecución del plan

- ¿Qué acciones toma?

Un Viernes en el Museo

- Sus respuestas probablemente se parezcan a las siguientes:

Requisitos

- ¿Qué pregunta Carlos en su mensaje de correo?
  - Si quiere ir al museo municipal el viernes a las 17:00

Diseñar un plan

- ¿Qué cosas debe tener en cuenta antes de salir?
  - Hay que acordar una hora (antes de las 17:00) para quedar en la estación de metro cerca del campus
  - Hay que echar un vistazo a un mapa de la ciudad y a un plano del metro

Pruebas

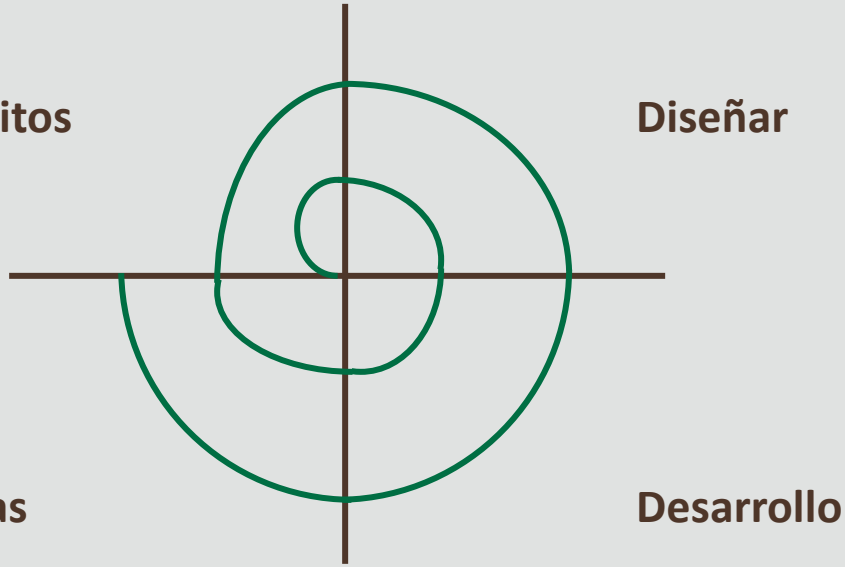
- ¿Cómo sabe que el plan funcionó?
  - ¿Se bajó en la parada derecha?
  - ¿Se llaman las calles y los edificios como esperaba?
  - ¿Ve alguna computadora?

Ejecución del plan

- ¿Qué acciones toma?
  - Tomo la línea roja del metro hasta la Estación del Sur
  - Cuando salgo de la estación, giro a la derecha y sigo recto hasta el final

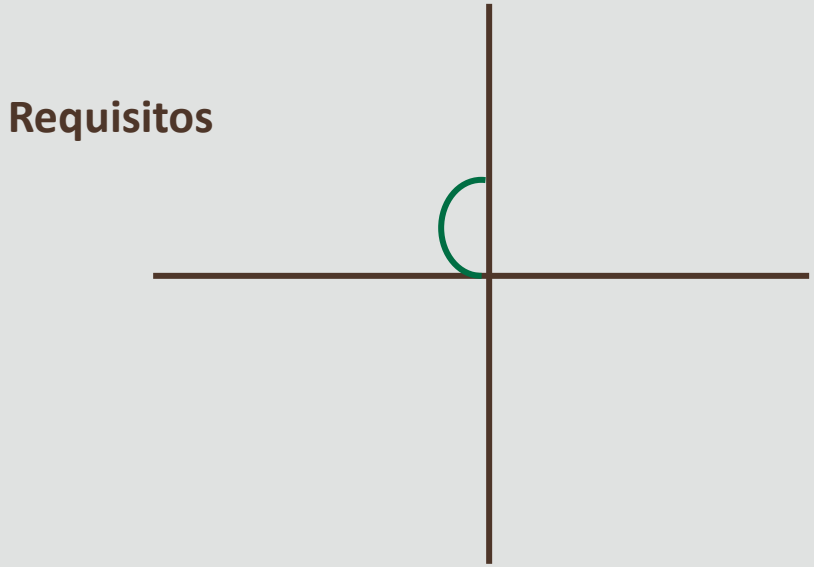
Introducción al Modelo Espiral de Desarrollo

- El desarrollo de software requiere seguir un proceso mental similar
- Esto se representa mediante el modelo espiral
- Existen otros modelos, pero el modelo espiral es el que mejor refleja lo que va a hacer en este curso



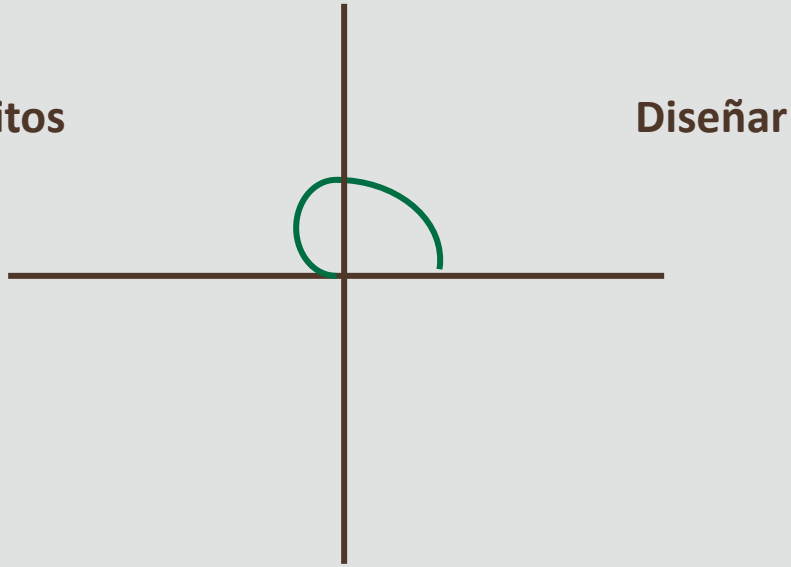
Requisitos

- Lea atentamente las instrucciones:
  - ¿Qué debe poder hacer su programa?
  - ¿Qué problemas pretende resolver?
  - ¿Qué funciones debe tener el programa?



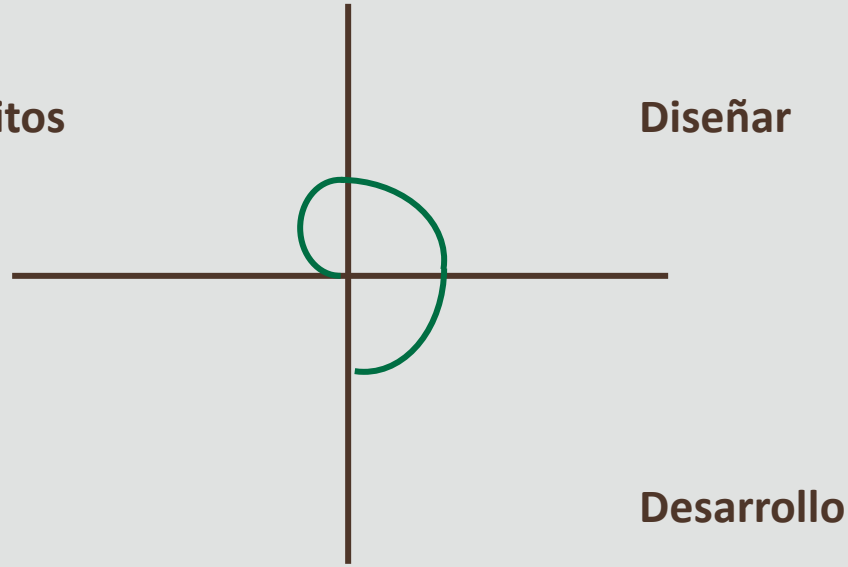
Diseñar

- Planifique el enfoque:
  - ¿Hay datos o comportamientos que el programa debe modelar?
  - ¿Hay alguna parte del programa que deba estar terminada antes de poder proseguir con otras partes?



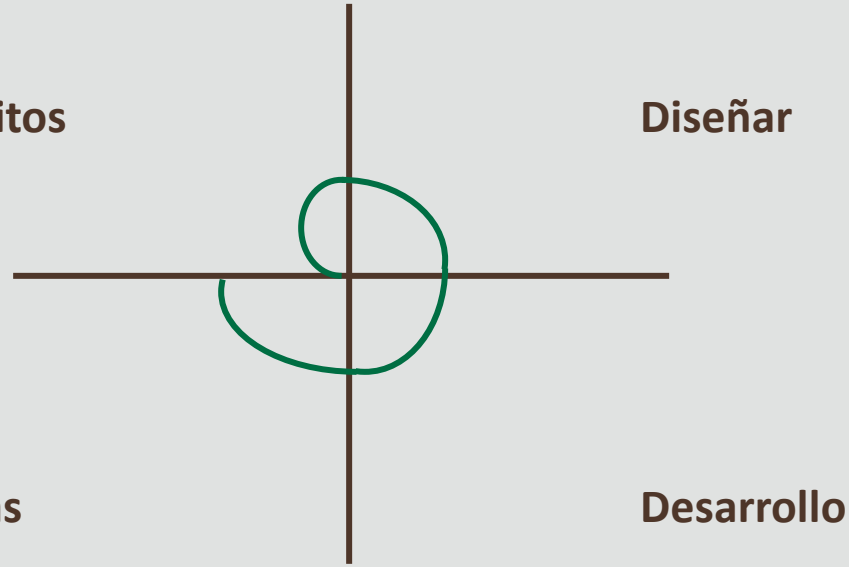
Desarrollo

- Empiece a codificar:
  - Cree una versión simplificada del programa
  - Céntrese en unas pocas funciones que sean sencillas o importantes



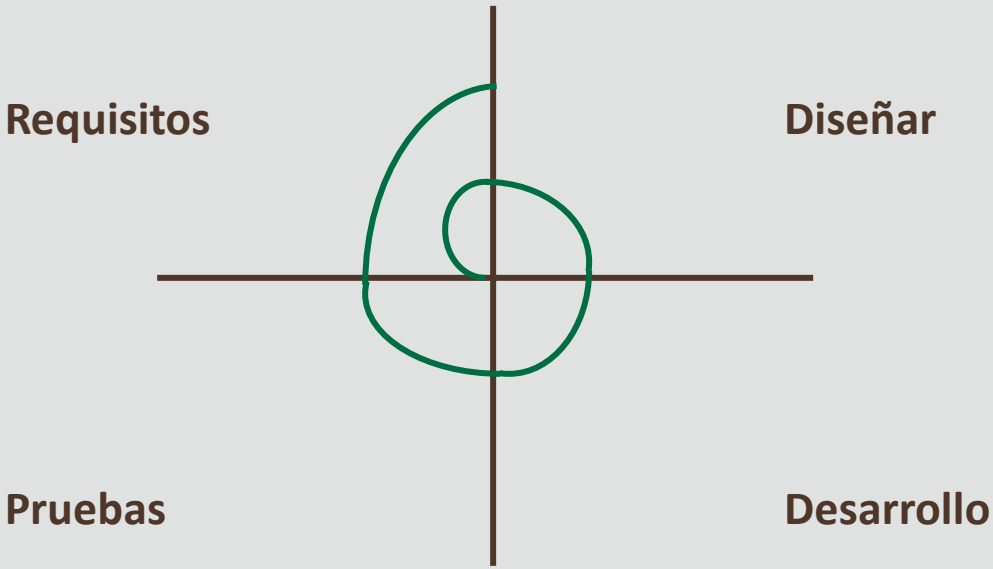
Test

- Pruebe su código:
  - ¿Está obteniendo los resultados que esperaba?
  - ¿Se han dado casos en los que se han obtenido resultados no deseados?
  - En función de su relevancia, puede que sea necesario corregir estos errores



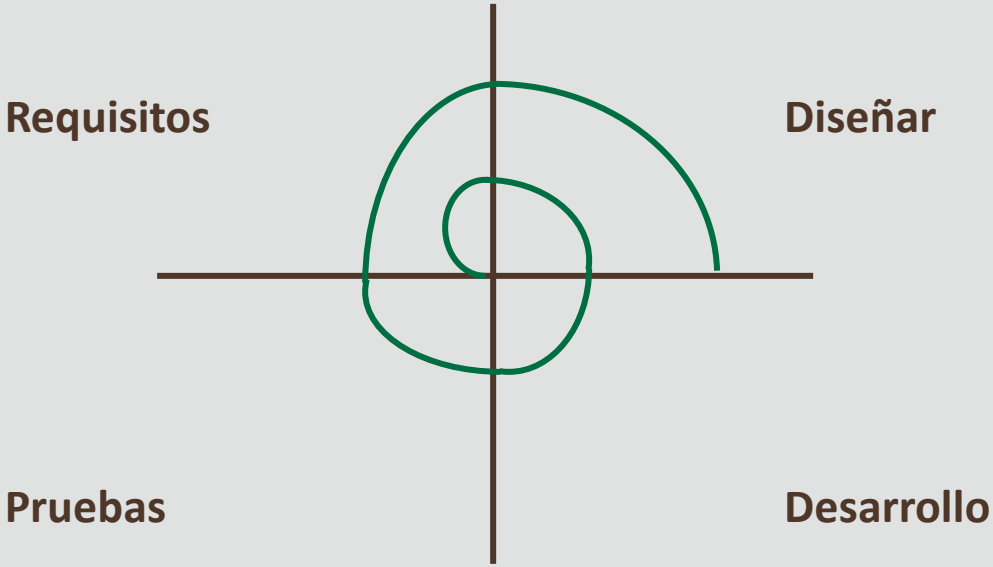
Iteración de los Requisitos

- Vuelva a comprobar los requisitos:
  - ¿Cumple el comportamiento del programa los requisitos?
  - ¿Hay requisitos o funciones adicionales que haya que incluir?
  - ¿Es necesario cambiar algún requisito?



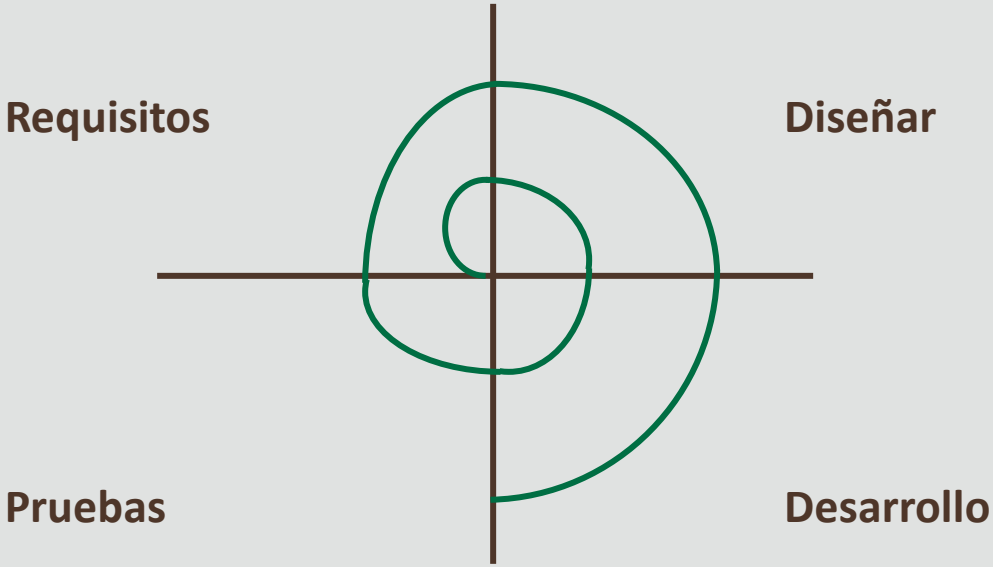
Iteración del diseño

- Planifique los cambios:
  - ¿Cómo se deben modelar las funciones adicionales?
  - ¿Es necesario cambiar el diseño para que se adapte mejor a las nuevas funciones que se van a añadir o a las funciones existentes que se van a ampliar?



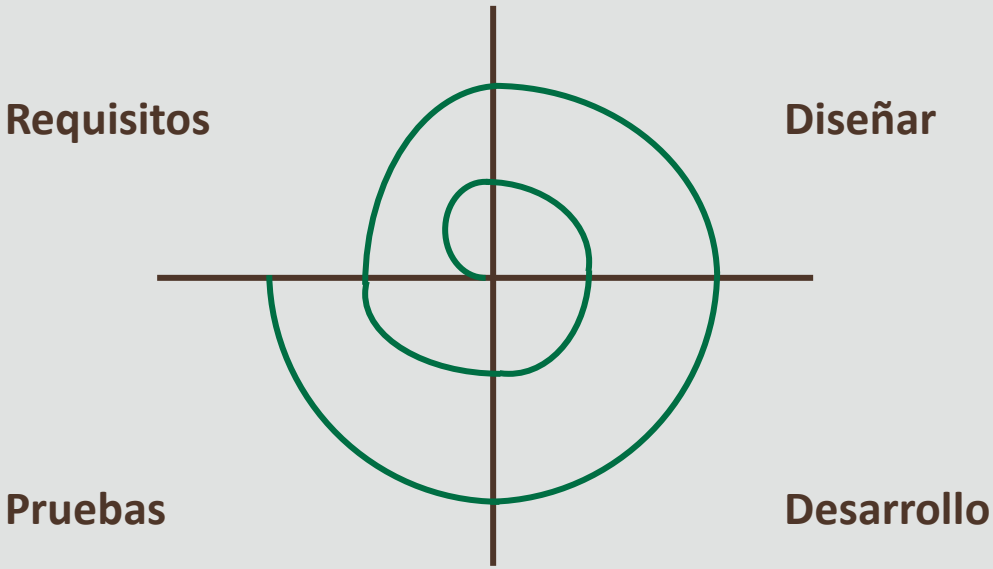
Iteración del desarrollo

- Continúe el proceso de desarrollo:
  - Añada nuevas funciones
  - Modifique o amplíe las funciones existentes, si es necesario



Más Pruebas

- Siga realizando pruebas:
  - ¿Funciona el nuevo código como esperaba?
  - ¿Sigue funcionando bien el código antiguo?
  - Es posible que haya que corregir algunos errores en función de la gravedad



Desarrollo, Pruebas y Correcciones

- El proceso de desarrollar, realizar pruebas, y corregir errores puede ser a veces frustrante:
  - El código a menudo no funciona
  - Aparecen errores inesperados
  - Las soluciones parecen difíciles de implementar

Programar es como Resolver Rompecabezas

- Puede que lleve mucho tiempo...
  - Pensar
  - Experimentar
  - Investigar e iterar
- Pero resulta muy gratificante...
  - Ver que el código por fin funciona (o se comporta ligeramente mejor)
  - Ver como el programa va evolucionando y volviéndose cada vez más robusto
  - Notar como cada vez va adquiriendo más soltura
  - Descubrir "sin querer" formas de generar errores

Recursos de Investigación

- Ha estado jugueteando un poco, pero ¿sigue sin saber qué hacer? Existen muchos recursos que pueden servirle de gran ayuda:
- Apuntes de clase y pequeños ejercicios que se hayan hecho
  - ¿Utilizan comandos o técnicas que le resultarían útiles?
- Documentación de Java de Oracle
  - En estos documentos se explican los comandos de Java disponibles.
  - https://docs.oracle.com/en/java/javase/17/docs/api/java.base/module-summary.html
- Internet
  - Es posible que otras personas ya hayan preguntado sus mismas dudas
  - Puede que encuentre ejemplos que le sirvan o comandos útiles que no conocía
  - Pero no se limite a copiar código; al fin y al cabo, la solución debe ser cosa suya

Ejercicio 2, Parte 1

- A continuación encontrará de nuevo el mensaje de correo de Carlos, por si lo necesita para hacer este ejercicio

Hola:

Al parecer, este mes hay una exposición sobre la historia de la informática en el museo municipal, y algunos del grupo estamos pensado en ir el viernes a las 17:00. ¿Te apuntas?
Creo que el metro sería la forma más rápida de llegar.

Carlos

Ejercicio 2, Parte 2

- Complete esta tabla
  - Imagínese que podría ocurrir si se le olvidara un paso concreto:

Requisitos	**Diseñar un plan**
Pruebas	**Ejecución del plan**

Un Viernes que Cayó en el Olvido

- Sus respuestas probablemente se parezcan a las siguientes:

Requisitos

- Acaba haciendo otra cosa el viernes

Diseñar un plan

- Todo el mundo ha cogido el tren, pero nadie sabe dónde va
- Lleva horas en el tren, pero nunca llega al muse

Pruebas

- Pasa el museo de largo
- El edificio al que ha llegado no es el museo
- El museo está cerrado

Ejecución del plan

- Aunque el plan era perfecto, nadie va al museo
- Carlos está triste

Olvidar Pasos en el Modelo Espiral

- Del mismo modo, las cosas pueden salir mal si se olvida uno de los pasos del modelo espiral

Requisitos

- El programa funciona, pero el problema que soluciona es otro diferente
- Faltan funciones

Diseñar

- El código es un lío
- Los errores son difíciles de solucionar
- Es difícil mejorar las funciones

Pruebas

- El programa sigue fallando de vez en cuando.
- Los resultados que se obtienen son incorrectos
- Los usuarios se sienten frustrados
- Los usuarios no pueden dejar de reír

Desarrollo

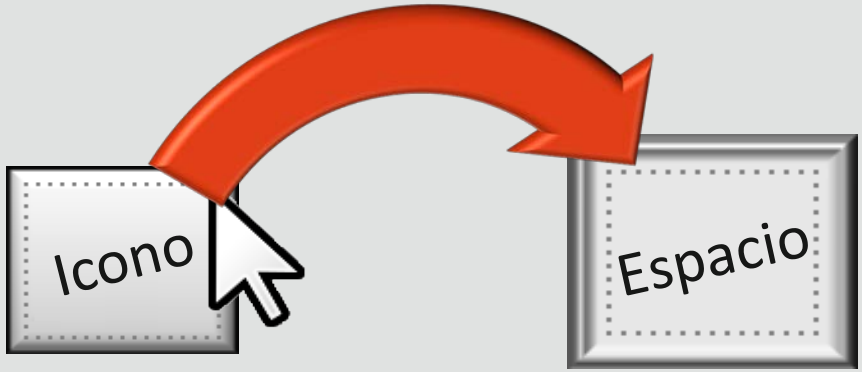
- No hay ningún programa

¿Qué es una Función de Software?

- Una función es:
  - Algo que un programa puede hacer
  - Algo que se puede hacer con un programa
- Ejemplos:
  - Imprimir texto
  - Reproducir un sonido
  - Calcular un valor
  - Arrastrar y soltar un icono
  - Publicar una puntuación alta en una tabla de clasificación online
  - Un nuevo tipo de enemigo en un videojuego

Implementación de una Función

- Algunas funciones son más fáciles de implementar que otras:
  - Se pueden codificar en unas pocas líneas sencillas
  - Por ejemplo, imprimir texto en la ventana de salida del IDE
- Algunas funciones son difíciles de implementar
  - Se basan en una combinación de varias funciones
  - Por ejemplo, poder "arrastrar y soltar" un icono

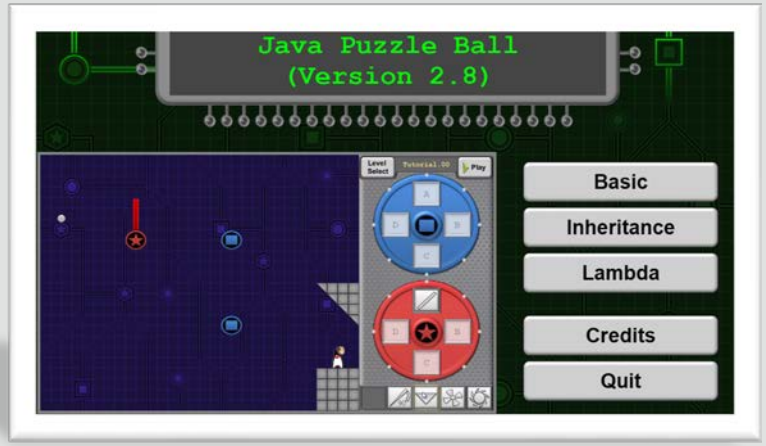


Implementación de la Función "Arrastrar y Soltar"

- La función "arrastrar y soltar" requiere varias funciones más pequeñas:
  - Añadir un gráfico en la pantalla
  - Encontrar la posición del mouse
  - Detectar el clic del mouse
  - Detectar cuando se suelta el mouse
  - Cambiar la posición del gráfico
- Implementar tan sólo uno de estos elementos puede suponer una gran satisfacción

Caso Práctico: Java Puzzle Ball

- Este juego está escrito completamente en Java FX
- Está diseñado con el fin de enseñar conceptos de programación
- Hemos guardado todas las versiones anteriores de este juego para que pueda explorar cómo se fueron implementando gradualmente las funciones



El Proceso de Desarrollo del Juego

- Estos son los pasos que intentamos seguir:
  - Hacer un foro de ideas y prototipos de ideas para el juego
  - Documentar los objetivos y requisitos de la mejor idea
  - Desglosar los requisitos en tareas/funciones y agregarlos a un programa
  - Desarrollar
  - Realice la prueba
  - Iterar y volver a evaluar los requisitos

¡Um!... Estos pasos me resultan familiares

Ejercicio 3, Parte 1

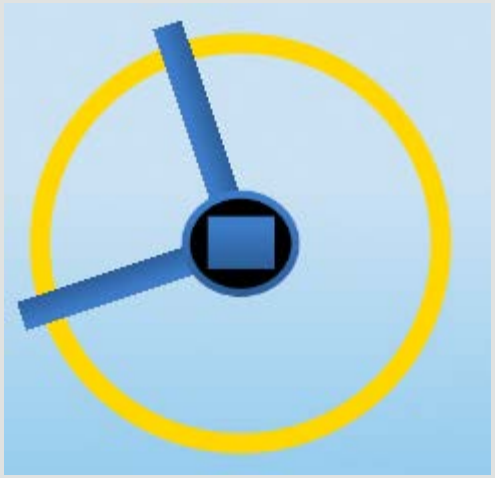
- Descargue, descomprima `OldGameVersions.zip` y reproduzca estas grabaciones de diferentes versiones del juego durante su desarrollo:
  - 16 de agosto de 2013 (08-16-13.mp4)
  - 22 de agosto de 2013 (08-22-13.mp4)
  - 27 de septiembre de 2013 (09-27-13.mp4)
  - 16 de octubre de 2013 (10-16-13.mp4)
  - 21 de noviembre de 2013 (11-21-13.mp4)

Ejercicio 3, Parte 2

- Dedique un par de minutos a explorar cada versión
- Observe las funciones nuevas, los errores y las modificaciones que difieren entre las distintas versiones

viernes, 16 de agosto de 2013

- Objetivos de esta versión:
  - Que el desarrollador aprenda a utilizar Java FX
  - Implementar algunas funciones básicas
- Funciones destacadas:
  - Mostrar imágenes en pantalla
  - Detectar eventos de mouse
  - Rotar los deflectores azules
  - Arrastrar y soltar un icono en los espacios (N, E)

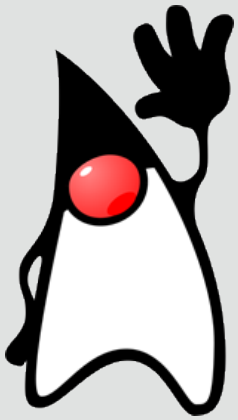


jueves, 22 de agosto de 2013

- Una semana más tarde:
  - Esta versión no es un juego todavía
  - Pero ya está teniendo mejor pinta
- Funciones destacadas:
  - Los iconos y las ruedas de la Interfaz de Usuario (UI) aparecen a la derecha
  - Un deflector rojo
  - Adjuntos en color
  - Más iconos para arrastrar y soltar

viernes, 27 de septiembre de 2013

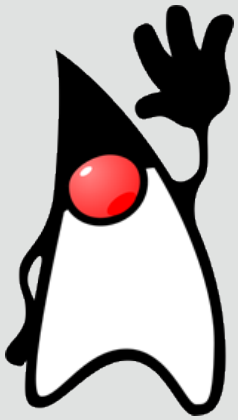
- Aproximadamente un mes más tarde:
  - Esta versión ya si podría considerarse un juego
  - El objetivo es desviar la bola a Duke
  - El código ha sido creado por otro desarrollador diferente



Duke

viernes, 27 de septiembre de 2013

- Funciones destacadas:
  - Un botón Play y un objetivo (Duke)
  - Una bola que se puede mover y desviar
  - Posibilidad de agregar más figuras
  - Líneas amarillas (para la detección de colisiones)
  - Ruedas que pasan al siguiente incremento de 45 grados



Duke

miércoles, 16 de octubre de 2013

- Unas semanas más tarde, hemos creado más modos de juego (Inheritance y Geometry Test)
- Hay una ventana emergente en la que se pueden seleccionar los niveles
  - Porque no sabíamos como descargar/cambiar niveles
  - Hay que cerrar el programa para carga otro nivel
  - Los niveles sirven para probar funciones; no son rompecabezas para los jugadores

miércoles, 16 de octubre de 2013

- Funciones más destacadas:
  - Geometría de nivel
  - Un deflector verde y una rueda verde
  - Las instrucciones para crear niveles se leen en formato de archivo de texto (pero no tenías por qué saberlo)

jueves, 21 de noviembre de 2013

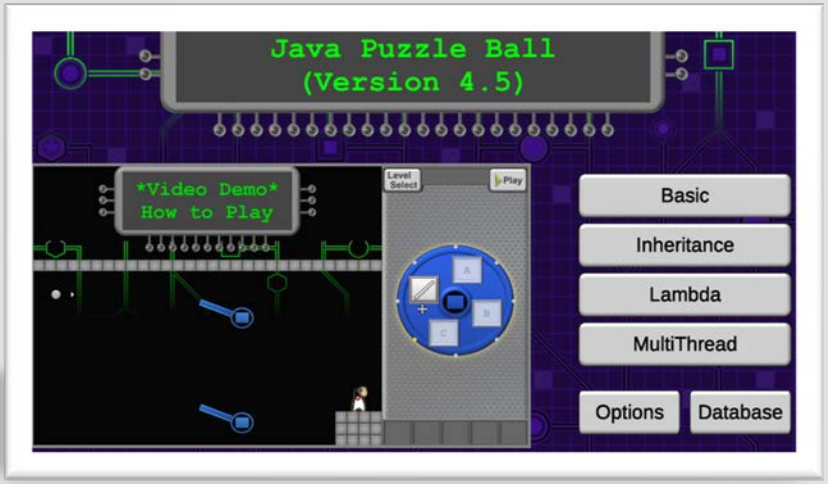
- Más de un mes después:
  - Hemos averiguado cómo descargar niveles
  - Solo se necesita un archivo para ejecutar el juego
- Utilice el botón Options para elegir el nivel
  - Se trata de una solución temporal hasta que aprendamos a crear menús
  - Los niveles son, de hecho, rompecabezas de verdad, en lugar de demostraciones técnicas

jueves, 21 de noviembre de 2013

- Funciones más destacadas:
  - Diseño de fondo más elaborado
  - Más niveles
  - Los espacios están marcados con ABCD en vez de NESW (porque la gente pensaba que, si el espacio N no estaba mirando hacia el Norte, era porque las soluciones eran erróneas)

Versión Actual

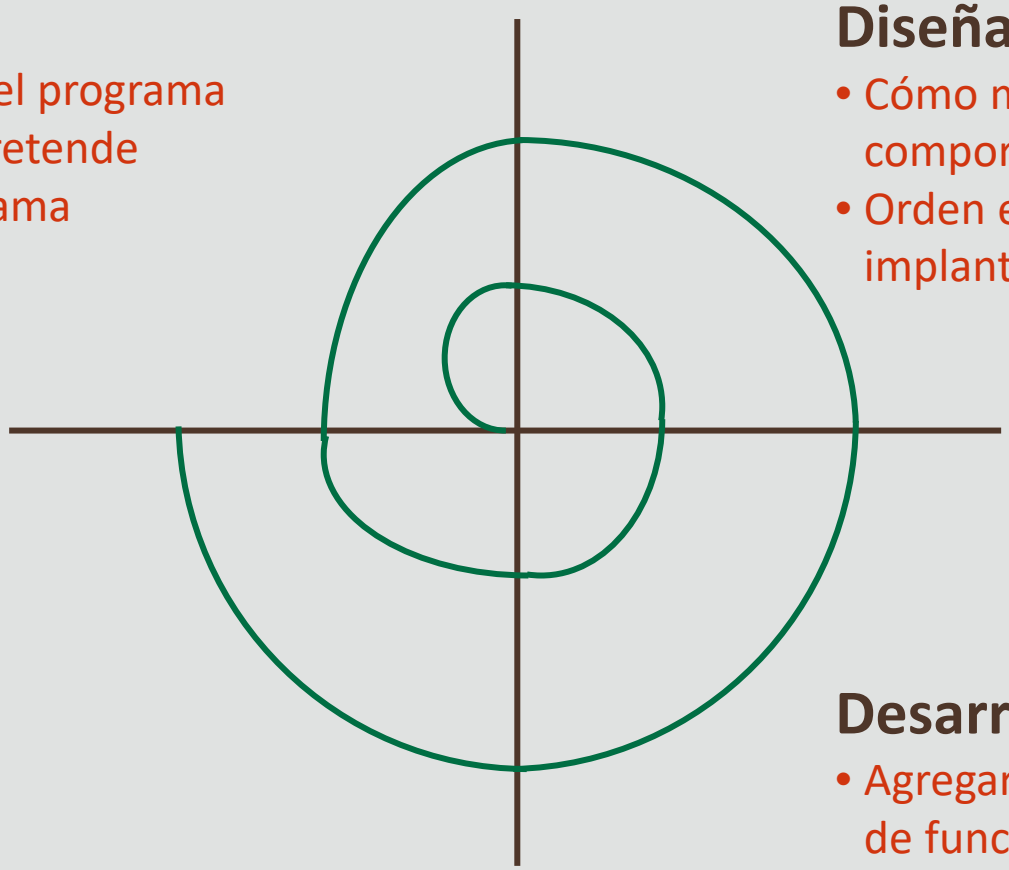
- El desarrollo continuó varios meses más en 2014 y se realizaron actualizaciones en 2020
- Observará que la última versión cuenta con nuevas funciones y modificaciones
- https://objectstorage.uk-london-1.oraclecloud.com/n/lrvrlgaqj8dd/b/Games/o/JavaPuzzleBall/index.html



Resumen del Modelo Espiral

Requisitos

- Qué debe hacer el programa
- Qué problema pretende resolver el programa



Diseñar

- Cómo modelar datos y comportamientos
- Orden en el que se deben implantar las funciones

Pruebas

- Buscar errores
- Corregir errores

Desarrollo

- Agregar versiones sencillas de funciones nuevas
- Mejorar las funciones existentes

Resumen

- En esta lección, debe haber aprendido lo siguiente:
  - Comprender el modelo espiral de desarrollo
  - Reconocer las tareas y subtareas del modelo espiral
  - Saber qué ocurre cuando se ignoran pasos
  - Identificar las funciones de software
  - Comprender cómo se implementan gradualmente las funciones


ORACLE
Academy