
ORACLE
Academy

Java Foundations

2-1

O Processo de Desenvolvimento do Software

ORACLE Academy

Objetivos

- Esta lição abrange os seguintes objetivos:
  - Entender o Modelo Espiral de desenvolvimento
  - Reconhecer tarefas e subtarefas do Modelo Espiral
  - Reconhecer o que acontece quando etapas são ignoradas
  - Identificar recursos do software
  - Entender como recursos são gradualmente implementados

Exercício 1, Parte 1

- Seu amigo, Clinton, tem planos para o fim de semana
- Leia o e-mail que ele enviou e considere quais etapas são necessárias para fazer com que esses planos aconteçam:

> *Olá, amigo,*
>
> *Vai ter uma apresentação especial da História do Computador no Museu da Cidade este mês. Estamos pensando em ir na sexta-feira às 17h. Você gostaria de ir conosco? Acho que o metrô seria a melhor opção para chegar lá.*
>
> *Clinton*

Exercício 1, Parte 2

- Complete o gráfico criando pelo menos um item para cada seção

Requisitos

- O que o e-mail de Clinton está perguntando?

Projetando um Plano

- O que você precisa considerar antes de sair?

Testando

- Como você sabe se o plano funcionou?

Implementando o Plano

- Quais ações você toma?

Sexta-feira no Museu

- Você deve ter escrito algo semelhante ao seguinte:

Requisitos

- O que o e-mail de Clinton está perguntando?
  - Estar no Museu da Cidade às 17h na sexta-feira

Projetando um Plano

- O que você precisa considerar antes de sair?
  - Marcar um horário de encontro na estação de metrô do campus antes das 17h
  - Consultar os mapas do metrô e das ruas

Testando

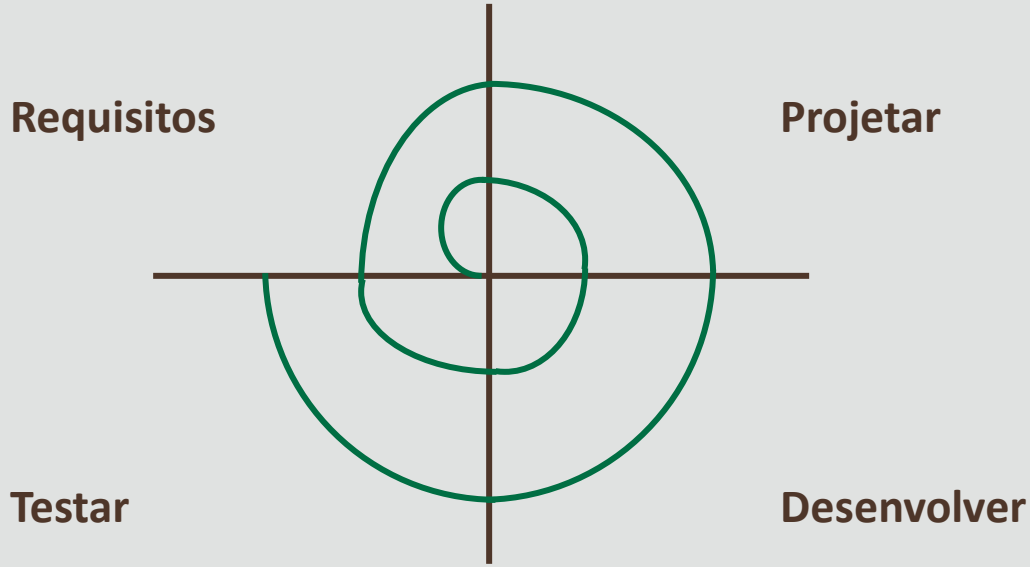
- Como você sabe se o plano funcionou?
  - Você desceu na estação correta?
  - Os nomes das ruas e dos prédios eram os que você esperava?
  - Você viu alguns computadores?

Implementando o Plano

- Quais ações você toma?
  - Pegue o metrô da linha vermelha em direção à South Station
  - Ande três quarteirões para leste

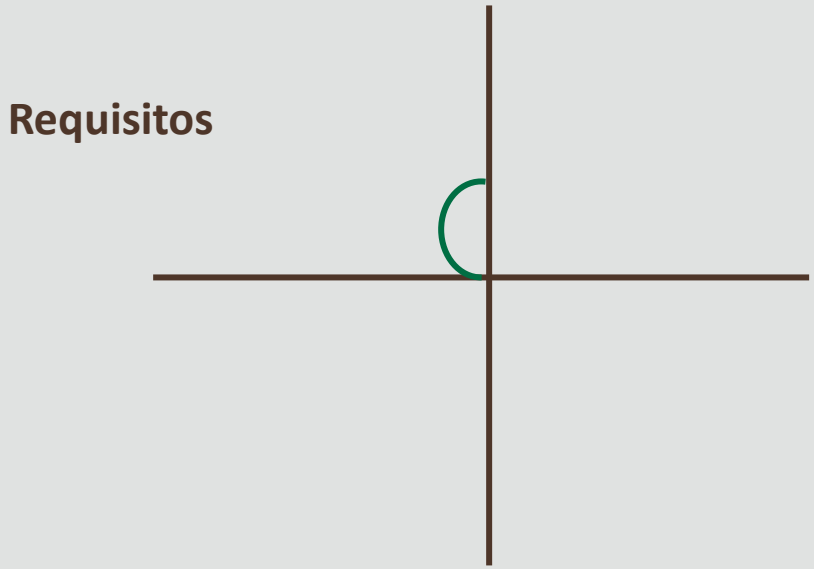
Apresentando o Modelo Espiral de Desenvolvimento

- O desenvolvimento de um software requer um processo de reflexão semelhante
- Isso é representado pelo Modelo Espiral
- Existem outros modelos, mas o Modelo Espiral é o que melhor reflete o que você estará fazendo neste curso



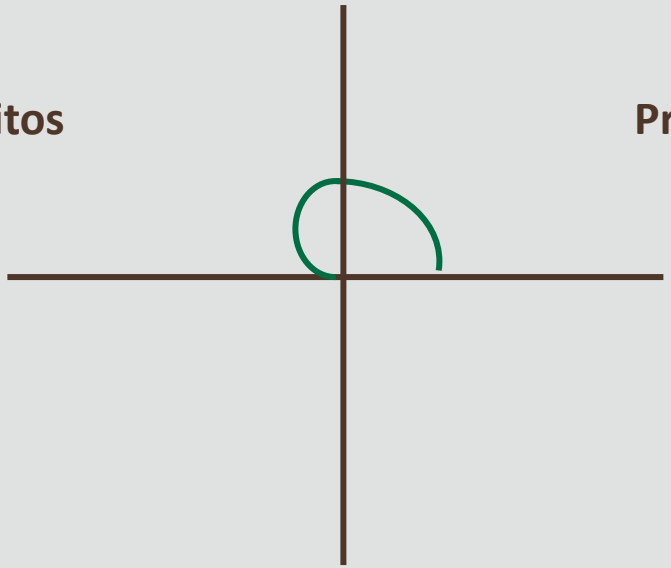
Requisitos

- Leia todas as instruções atentamente:
  - O que seu programa deve fazer?
  - Que problemas ele está tentando resolver?
  - Que recursos seu programa deve ter?



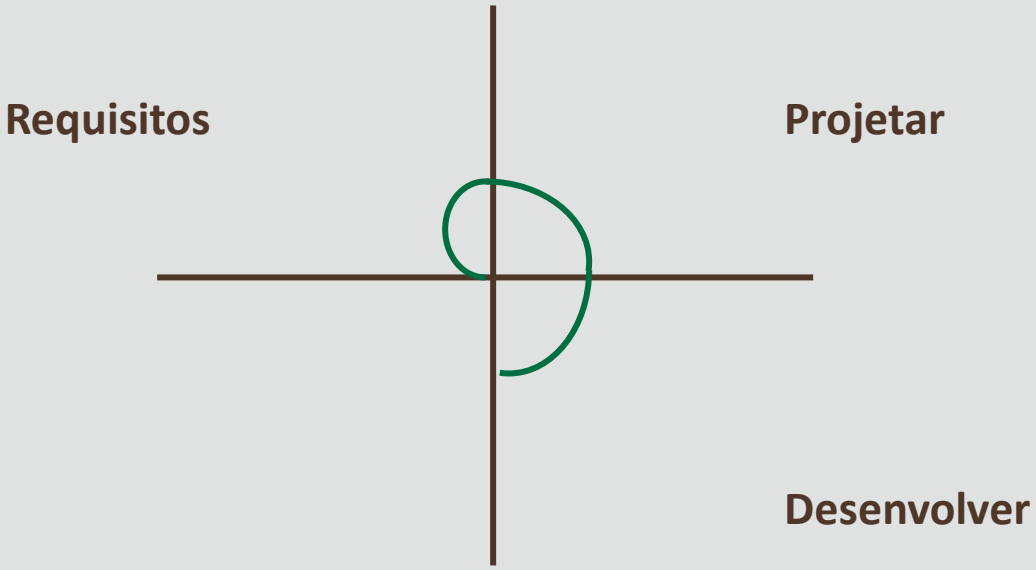
Projetar

- Planeje sua abordagem:
  - Seu programa precisa modelar dados ou comportamentos?
  - Partes específicas do programa precisam ser concluídas antes de você poder iniciar outras partes?



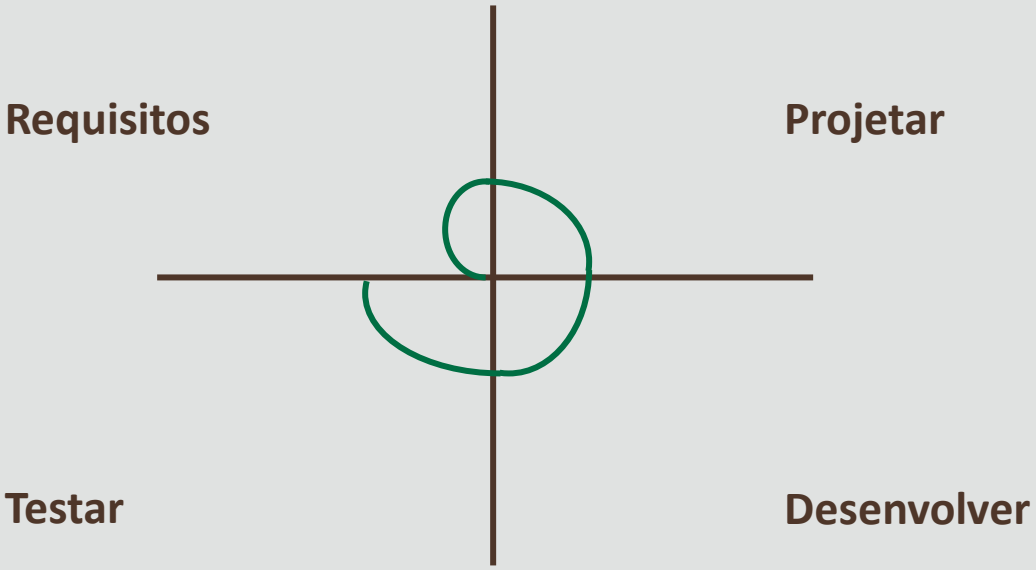
Desenvolver

- Inicie a codificação:
  - Crie uma versão simplificada do seu programa
  - Foque em um número pequeno de recursos simples ou importantes



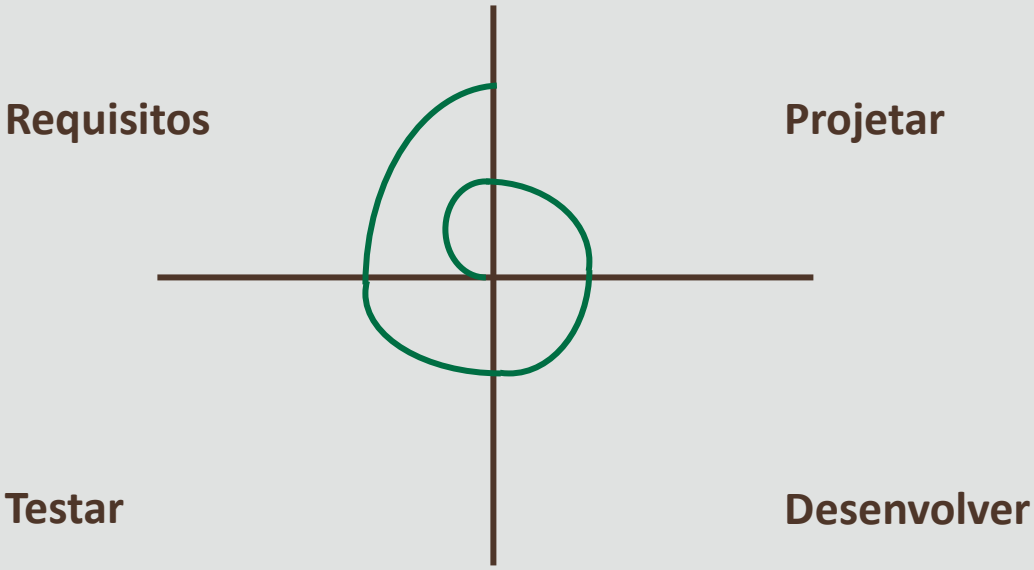
Testar

- Teste seu código:
  - O programa fornece os resultados esperados?
  - Algum cenário produz resultados indesejados?
  - Dependendo do impacto, esses bugs podem precisar ser corrigidos



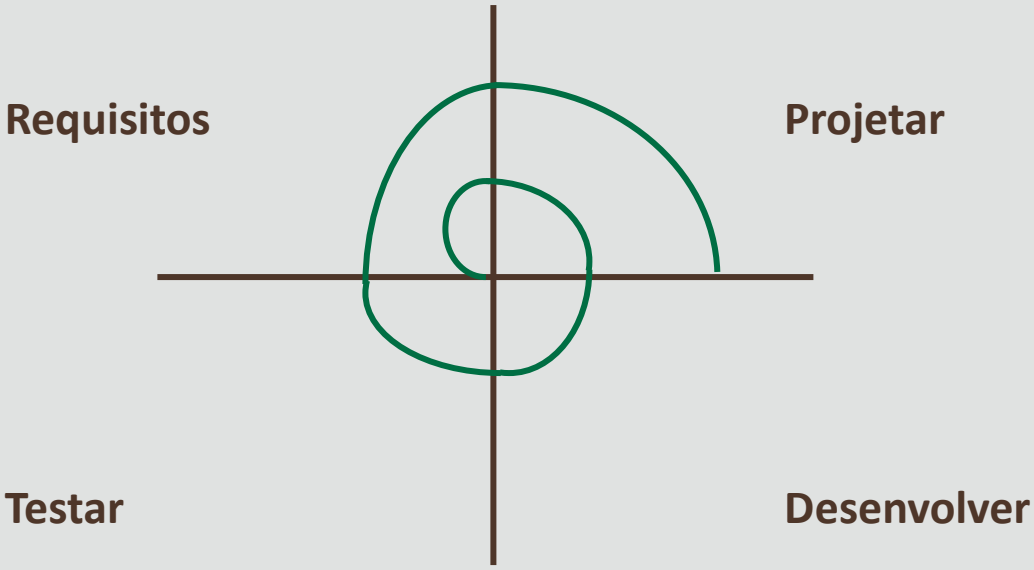
Iteração de Requisitos

- Verifique os requisitos novamente:
  - O comportamento do programa corresponde aos requisitos?
  - Existem requisitos ou recursos adicionais para serem construídos?
  - Alguns requisitos precisam ser alterados?



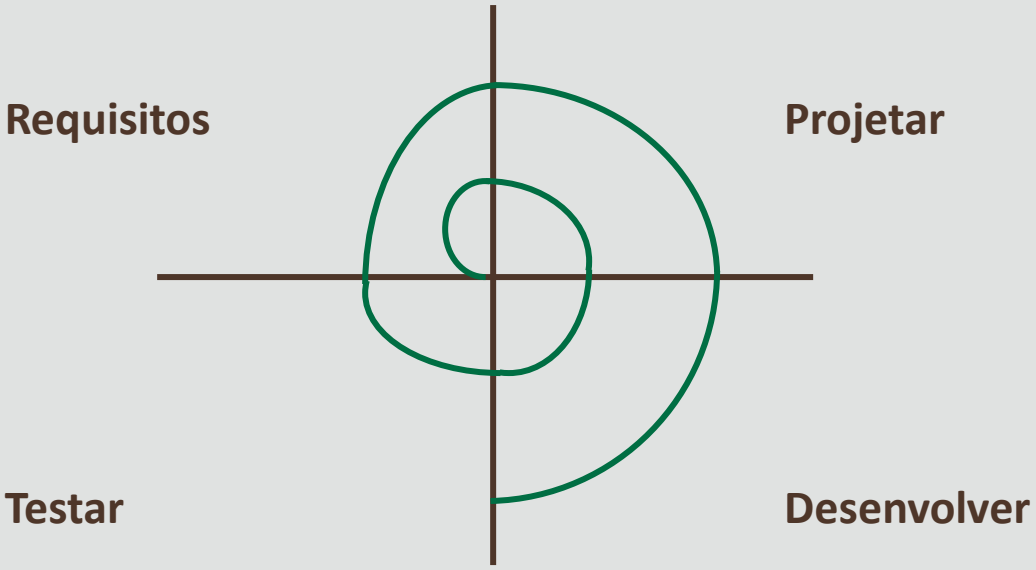
Iteração do Design

- Planeje suas alterações:
  - Como você deve modelar recursos adicionais?
  - É necessário alterar o design atual para suportar melhor a expansão dos recursos atuais ou para adicionar novos recursos?



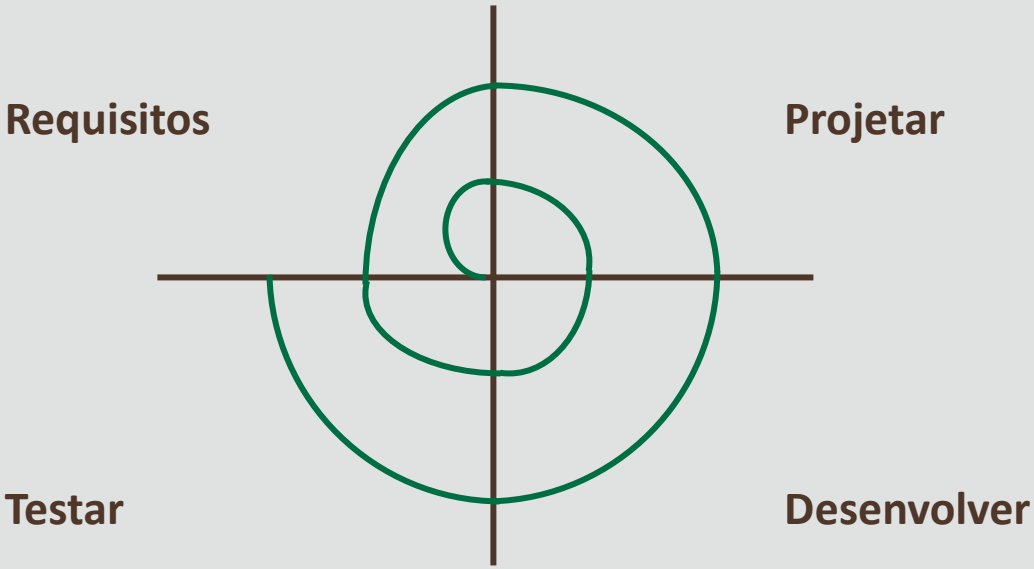
Iteração do Desenvolvimento

- Continue desenvolvendo:
  - Adicione novos recursos
  - Se necessário, modifique ou aprimore os recursos existentes



Mais Testes

- Continue testando:
  - O novo código funciona como você esperava?
  - O código antigo continuará a funcionar corretamente?
  - Dependendo da severidade, os bugs precisam ser corrigidos



Desenvolvendo, Testando e Corrigindo

- O processo de desenvolvimento, teste e correção de bugs às vezes é frustrante:
  - Normalmente, o código não funciona
  - Bugs inesperados aparecem
  - As soluções parecem difíceis e evasivas

Programar é como Montar Quebra-cabeças

- Pode levar algum tempo...
  - Pensando
  - Experimentando
  - Pesquisando e repetindo
- Mas é muito gratificante...
  - Ver seu código finalmente funcionando (ou se comportando um pouco melhor)
  - Ver seu programa evoluir e tornar-se mais robusto
  - Perceber que você está mais habilidoso
  - Encontrar maneiras divertidas de produzir bugs

Como Pesquisar

- Você continua confuso depois de tantas correções? Existem muitos recursos para ajudar você a progredir:
- Anotações das aulas e exercícios rápidos já feitos
  - Eles usam os comandos e as técnicas que você está procurando?
- Documentação do Oracle Java
  - Descreve os comandos Java disponíveis
  - https://docs.oracle.com/en/java/javase/17/docs/api/java.base/module-summary.html
- Internet
  - Pode ser que outras pessoas tenham feito perguntas semelhantes às suas
  - Você pode descobrir exemplos úteis ou novos comandos
  - Mas suas soluções devem ser de sua autoria, e não um código copiado

Exercício 2, Parte 1

- Veja a seguir novamente o e-mail que Clinton enviou, caso você precise consultá-lo neste exercício

> *Olá, amigo,*
>
> *Vai ter uma apresentação especial da História do Computador no Museu da Cidade este mês. Estamos pensando em ir na sexta-feira às 17 h. Você gostaria de ir conosco? Acho que o metrô seria a melhor opção para chegar lá.*
>
> *Clinton*

Exercício 2, Parte 2

- Complete este gráfico
  - Imagine o que aconteceria com a sua noite no museu se uma etapa específica fosse esquecida:

Sexta-feira Esquecida

- Você deve ter escrito algo semelhante ao seguinte:

Requisitos

- Você já tem compromisso na sexta-feira

Projetando um Plano

- Todo mundo está no metrô, mas ninguém sabe para onde está indo
- Você fica horas andando de metrô, mas nunca chega ao museu

Testando

- Você passa pelo museu
- Você chega no edifício errado
- O museu está fechado

Implementando o Plano

- Apesar de ser um plano incrível, ninguém vai ao museu
- Clinton está chateado

ORACLE Academy

Esquecendo Etapas no Modelo Espiral

- Da mesma forma, coisas ruins podem acontecer quando determinada etapa do Modelo Espiral é esquecida

Requisitos

- O programa funciona, mas não resolve o problema correto
- Estão faltando recursos

Projetar

- O código está desorganizado
- Os bugs são difíceis de serem corrigidos
- Os recursos são difíceis de serem aprimorados

Teste

- O programa continua travando
- O programa fornece resultados incorretos
- Os usuários ficam frustrados
- Os usuários não param de rir

Desenvolvimento

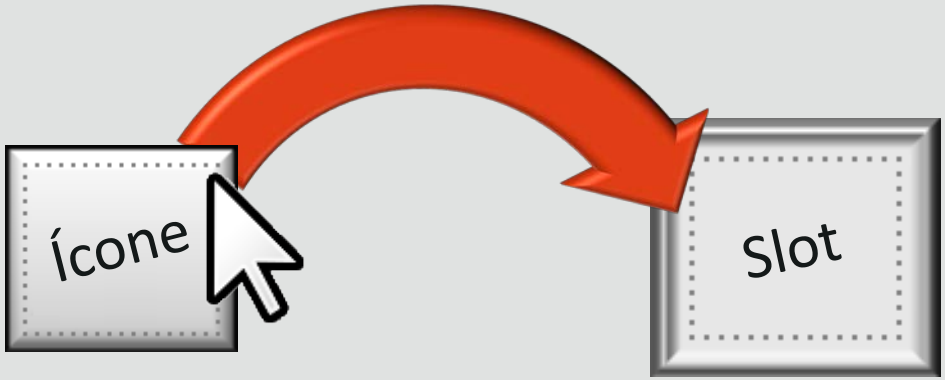
- Não há um programa

O que É um Recurso de Software?

- Pense em um recurso como:
  - Algo que um programa pode fazer
  - Algo que você pode fazer com um programa
- Exemplos:
  - Imprimir um texto
  - Reproduzir um som
  - Calcular um valor
  - Arrastar e soltar um ícone
  - Dar uma pontuação alta em uma classificação semanal on-line
  - Um novo tipo de inimigo em um videogame

Implementando um Recurso

- Alguns recursos são mais fáceis de serem implementados:
  - Você pode codificá-los em algumas linhas simples
  - Por exemplo, imprimir um texto na janela de saída do seu IDE
- Alguns recursos são difíceis de serem implementados
  - Eles baseiam-se em uma combinação de outros recursos
  - Por exemplo, ser capaz de "arrastar e soltar" um ícone

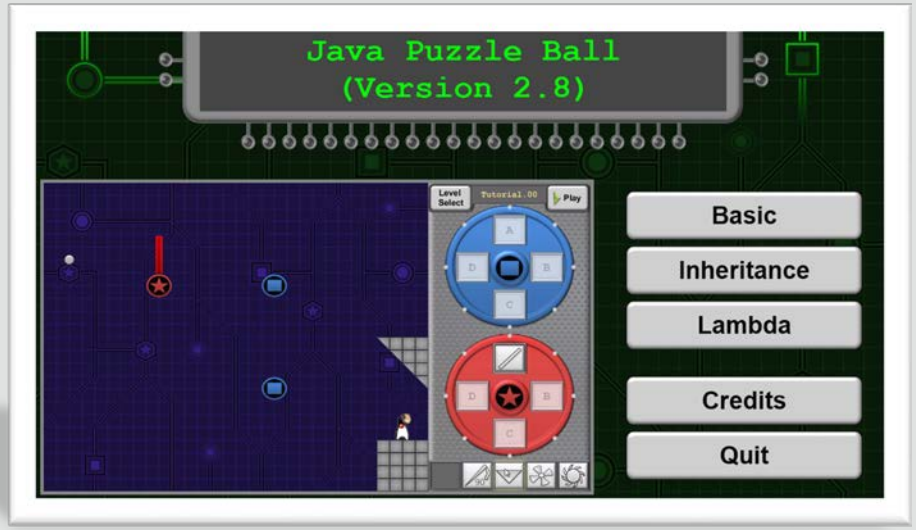


Implementando o Recurso "Arrastar e Soltar"

- Um recurso "arrastar e soltar" requer vários recursos menores:
  - Adicionar um gráfico à tela
  - Encontrar a posição do mouse
  - Detectar um clique no botão do mouse
  - Detectar a liberação do botão do mouse
  - Alterar a posição do gráfico
- A implementação de apenas um desses itens pode parecer uma grande conquista

Estudo de Caso: Java Puzzle Ball

- Este jogo é totalmente escrito em Java FX
- Ele é projetado para ensinar conceitos de programação
- Salvamos todas as versões antigas do jogo para que você consiga explorar o modo como os recursos foram gradualmente implementados!



O Processo de Desenvolvimento do Jogo

- Estas são as etapas que tentamos executar:
  - Coletar e elaborar as ideias do jogo
  - Documentar metas e requisitos para a melhor ideia
  - Dividir os requisitos em tarefas/recursos e adicioná-los a um cronograma
  - Desenvolver
  - Testar
  - Repetir e reavaliar os requisitos

Hmm... Essas etapas parecem familiares

Exercício 3, Parte 1

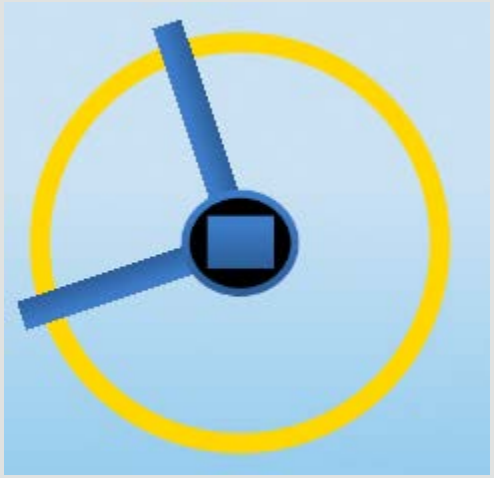
- Faça download do arquivo `OldGameVersions.zip`, descompacte-o e reproduza essas gravações das diferentes versões do game durante o desenvolvimento:
  - 16 de agosto de 2013 (08-16-13.mp4)
  - 22 de agosto de 2013 (08-22-13.mp4)
  - 27 de setembro de 2013 (09-27-13.mp4)
  - 16 de outubro de 2013 (10-16-13.mp4)
  - 21 de novembro de 2013 (11-21-13.mp4)

Exercício 3, Parte 2

- Dedique alguns minutos para explorar cada nova versão
- Observe cada novo recurso, bug ou alteração entre as versões

16 agosto de 2013

- As metas desta versão:
  - Fazer com que o desenvolvedor aprenda sobre Java FX
  - Implementar alguns recursos básicos
- Recursos em destaque:
  - Exibir imagens na tela
  - Detectar eventos do mouse
  - Girar BlueBumpers
  - Arrastar e soltar um ícone em slots (N, E)

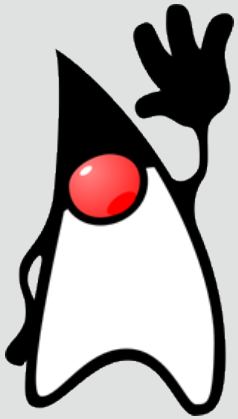


22 agosto de 2013

- Uma semana depois:
  - Esta versão ainda não é um jogo
  - Mas está ficando mais interessante
- Recursos em destaque:
  - Discos e ícones da interface do usuário posicionados à direita
  - Um RedBumper
  - Anexos coloridos
  - Mais ícones para arrastar e soltar

27 de setembro de 2013

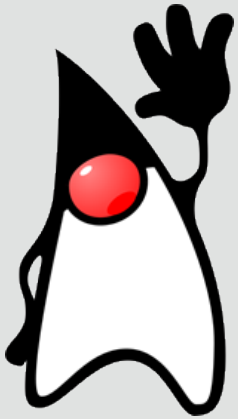
- Cerca de um mês depois:
  - Esta versão já poderia ser chamada de jogo
  - A meta é desviar a bola para o Duke
  - Um outro desenvolvedor criou o código



Duke

27 de setembro de 2013

- Recursos em destaque:
  - Um botão Reproduzir e uma meta (Duke)
  - Uma bola capaz de se mover e ser desviada
  - Mais formas que podem ser anexadas
  - Linhas amarelas (para detecção de colisão)
  - Discos que se aproximam do incremento mais perto de 45 graus



Duke

16 de outubro de 2013

- Algumas semanas depois, criamos alguns modos adicionais do jogo (Herança e Teste de Geometria)
- Existe um pop-up para escolher níveis
  - Porque não sabíamos como mudar/alternar entre os níveis
  - Você precisa fechar o programa para carregar outro nível
  - Os níveis são para testar os recursos e não são tão desafiadores para os jogadores

16 de outubro de 2013

- Mais recursos em destaque:
  - Geometria de nível
  - GreenBumper e GreenWheel
  - Instruções sobre criação de níveis são lidas em um arquivo de texto, mas você não tinha como saber isso

21 de novembro de 2013

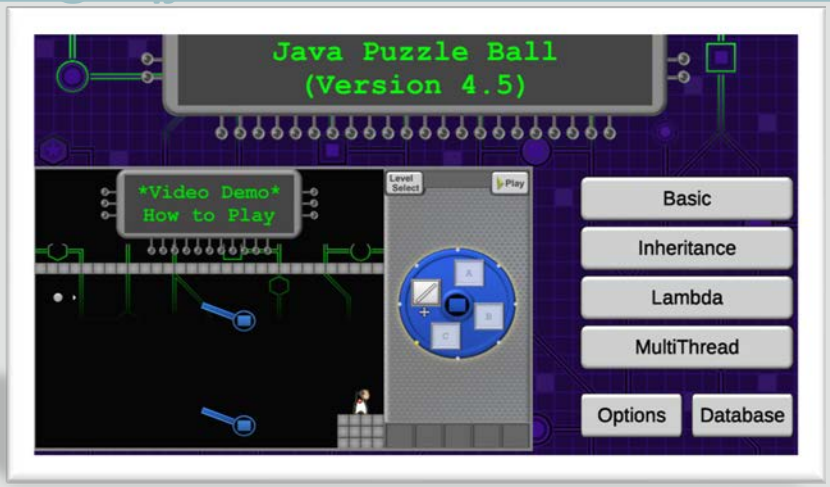
- Mais de um mês depois:
  - Descobrimos como passar de níveis!
  - É necessário um único arquivo para executar o jogo
- Use o botão Opções para escolher níveis
  - Essa é uma solução temporária até aprendermos a criar menus
  - Os níveis são quebra-cabeças reais, e não demonstrações técnicas

21 de novembro de 2013

- Mais recursos em destaque:
  - Uma nova arte de plano de fundo mais sofisticada
  - Mais níveis
  - Os slots são identificados como ABCD, e não como NESW (as pessoas pensariam que as soluções estariam erradas se o N não apontasse para o norte)

A Versão Atual

- O desenvolvimento continuou por vários meses mais durante 2014, e atualizações foram feitas em 2020
- Você perceberá novos recursos e mudanças na versão mais recente
- https://objectstorage.uk-london-1.oraclecloud.com/n/lrvrlgaqj8dd/b/Games/o/JavaPuzzleBall/index.html



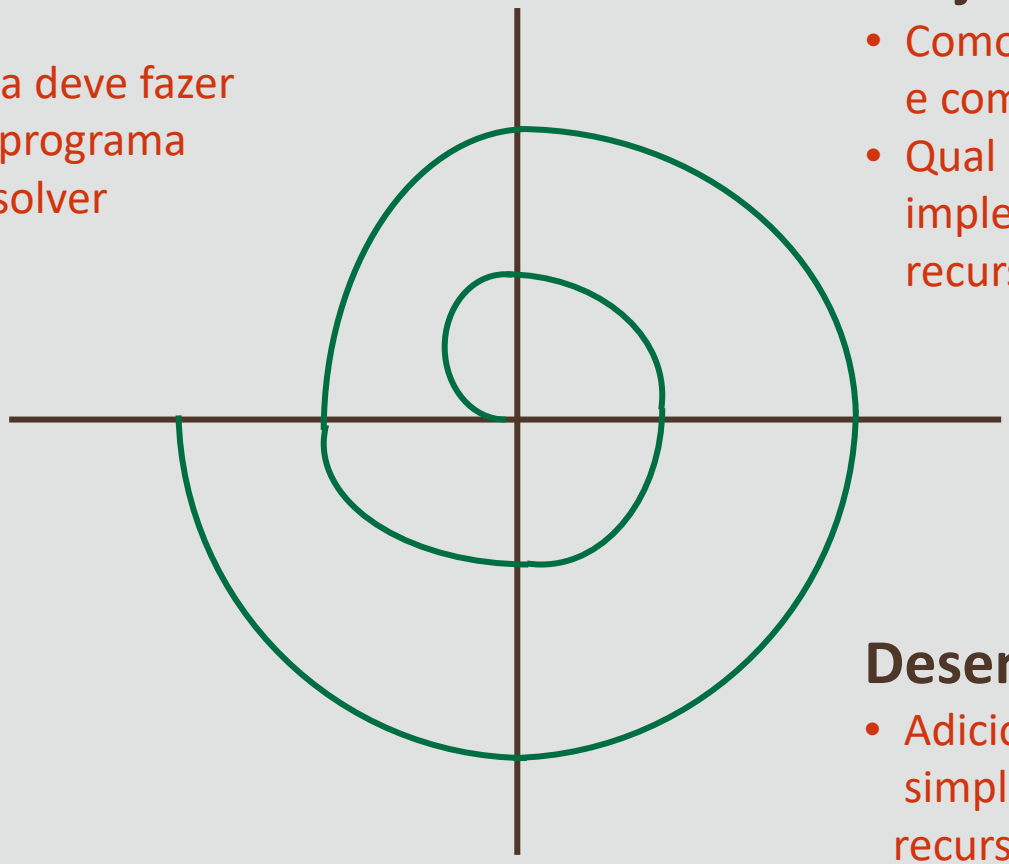
Resumo do Modelo Espiral

Requisitos

- O que o programa deve fazer
- Que problema o programa está tentando resolver

Projetar

- Como modelar dados e comportamentos
- Qual é a ordem de implementação dos recursos



Testar

- Encontrar bugs
- Corrigir bugs

Desenvolver

- Adicionar versões simples de novos recursos
- Melhorar recursos existentes

ORACLE Academy

Resumo

- Nesta lição, você deverá ter aprendido a:
  - Entender o Modelo Espiral de desenvolvimento
  - Reconhecer tarefas e subtarefas do Modelo Espiral
  - Reconhecer o que acontece quando etapas são ignoradas
  - Identificar recursos do software
  - Entender como recursos são gradualmente implementados

ORACLE Academy


ORACLE
Academy