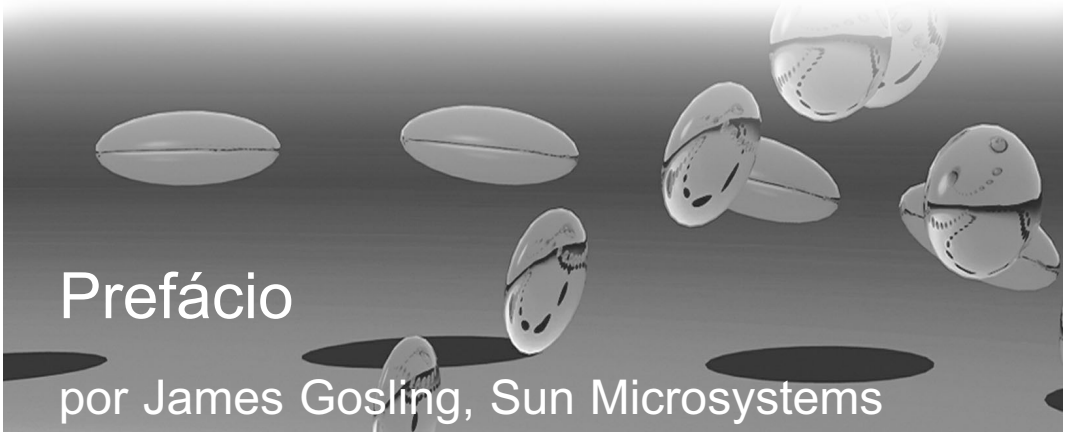

Prefácio

por James Gosling, Sun Microsystems

Observar minha filha Kate e seus colegas de faculdade, fazer um curso Java utilizando um IDE comercial era uma experiência terrível. A sofisticação da ferramenta adicionava complexidade significativa à tarefa de aprendizagem. Olhando para trás, gostaria de ter podido entender mais cedo o que estava acontecendo. Mas, do jeito que as coisas ocorreram, não fui capaz de conversar com o instrutor sobre o problema até que fosse tarde demais. Esse é exatamente o tipo de situação para a qual o BlueJ é perfeitamente adequado.

O BlueJ é um ambiente interativo de desenvolvimento com uma missão: foi projetado para ser utilizado por alunos que estão aprendendo a programar. Foi projetado por instrutores que estiveram na sala de aula defrontando-se com esse problema todos os dias. Foi revigorante falar com as pessoas que desenvolveram o BlueJ: elas têm uma idéia muito clara de qual é o seu alvo. As discussões tendiam a enfocar mais o que se deveria omitir do que o que se deveria discutir. O BlueJ está muito limpo e é muito objetivo.

Não obstante, este livro não é sobre o BlueJ. É sobre programação.

Em Java.

Nos últimos anos a programação em Java tem sido amplamente utilizada no ensino de programação.

Isso ocorre por muitas razões. Uma é que Java tem muitas características que facilitam o ensino: ela tem uma definição relativamente limpa; a extensa análise estática do compilador informa os alunos sobre eventuais problemas cedo no processo de desenvolvimento; e possui um modelo de memória muito robusto que elimina a maioria dos 'misteriosos' erros que surgem quando os limites dos objetos ou o sistema de tipos são comprometidos. A outra é que Java tornou-se comercialmente muito importante.

Este livro encara de frente o conceito mais difícil de ensinar: objetos. Ele conduz os alunos durante seus primeiros passos até alguns conceitos muito sofisticados.

Ele conduz as coisas de modo a resolver uma das perguntas mais ardilosas ao escrever um livro sobre programação: como lidar com a técnica de realmente escrever e executar um programa. A maioria dos livros pula discretamente a questão ou a aborda superficialmente, deixando para o instrutor descobrir como resolver o problema. Além de deixar o instrutor com o fardo de relacionar o material a ser ensinado com os passos que os alunos têm de dar para trabalhar com os exercícios. Este livro, porém, assume a utilização do BlueJ e é capaz de integrar as tarefas de entender os conceitos com a técnica de como os alunos podem explorá-los.

Eu gostaria que ele estivesse disponível para minha filha no ano passado. Talvez no ano que vem...

Prefácio para o instrutor

Este livro é uma introdução à programação orientada a objetos para iniciantes. O foco principal do livro são conceitos gerais de orientação a objetos e programação sob uma perspectiva de engenharia de software.

Enquanto os primeiros capítulos são escritos para alunos sem experiência em programação, os capítulos finais são adequados para programadores mais avançados ou profissionais. Em particular, programadores com experiência em uma linguagem não-orientada a objetos que desejam migrar suas habilidades para a orientação a objeto também devem ser capazes de se beneficiar do livro.

Utilizamos duas ferramentas por todo o livro para permitir que os conceitos introduzidos fossem colocados em prática: a linguagem de programação Java e o ambiente de desenvolvimento Java BlueJ.

Java

Java foi escolhida por causa de uma combinação de dois aspectos: o projeto da linguagem e sua popularidade. A própria linguagem de programação Java fornece uma implementação muito limpa da maioria dos conceitos orientados a objetos importantes e serve bem como uma linguagem de ensino introdutória. Sua popularidade assegura um grupo imenso de recursos de suporte.

Em qualquer área de assunto, ter várias fontes de informações disponíveis é muito útil, para professores e alunos. Para Java, em particular, incontáveis livros, tutoriais, exercícios, compiladores, ambientes e questionários já existem, em muitos tipos e estilos diferentes. Muitos deles estão on-line e muitos estão disponíveis gratuitamente. A grande quantidade e boa qualidade de material de suporte faz do Java uma excelente escolha como uma introdução à programação orientada a objetos.

Com tanto material sobre Java já disponível, há ainda lugar para dizer mais? Acreditamos que sim e a segunda ferramenta que utilizamos é uma das razões ...

BlueJ

A segunda ferramenta, o BlueJ, merece mais comentário. Este livro é único em sua utilização completamente integrado do ambiente do BlueJ.

O BlueJ é um ambiente de desenvolvimento Java que foi projetado na Monash University, Austrália, explicitamente como um ambiente para ensinar programação introdutória orientada a objetos. Ele é mais adequado ao ensino introdutório do que outros ambientes por várias razões:

- A interface com o usuário é muito mais simples. Alunos iniciantes podem geralmente utilizar o ambiente BlueJ de maneira apropriada depois de 20 minutos de introdução. A partir daí, a instrução pode se concentrar nos conceitos importantes à mão – a orientação ao objeto e Java –, e nenhum tempo precisa ser desperdiçado falando sobre ambientes, sistemas de arquivos, caminhos de classe, comandos do DOS ou conflitos de DLL.
- O ambiente suporta importantes ferramentas de ensino não-disponíveis em outros ambientes. Uma deles é a visualização de estrutura de classe. O BlueJ automaticamente exibe um diagrama UML que representa as classes e relacionamentos em um projeto. A visualização desses conceitos importantes é uma grande ajuda tanto para professores quanto para alunos. É difícil assimilar o conceito de um objeto quando tudo o que você vê na tela são linhas de código! A notação de diagrama é um subconjunto simples de UML, novamente personalizado de acordo com as necessidades dos alunos iniciantes. Isso facilita a compreensão, mas também permite migração para toda a UML em cursos posteriores.
- Uma das forças mais importantes do ambiente do BlueJ é a capacidade de o usuário criar diretamente objetos de qualquer classe e então interagir com seus métodos. Isso cria a oportunidade para experimentação direta com objetos, com pouco overhead no ambiente. Os alunos podem quase 'sentir' o que significa criar um objeto, chamar um método, passar um parâmetro ou receber um valor de retorno. Eles podem experimentar um método imediatamente depois de escrevê-lo, sem a necessidade de escrever drivers de teste. Essa facilidade é uma ajuda inestimável no entendimento dos detalhes subjacentes dos conceitos e da linguagem.

O BlueJ é um ambiente Java completo. Não é uma versão reduzida, simplificada de Java para ensino. Ele executa por cima do Java Development Kit da Sun Microsystems e utiliza o compilador-padrão e a máquina virtual. Isso assegura que ele sempre obedeça à especificação Java oficial e mais atualizada.

Os autores deste livro têm vários anos de experiência no ensino do ambiente BlueJ (e muitos mais anos sem ele antes disso). Na nossa experiência, testemunhamos quanto a utilização do BlueJ aumentou o envolvimento, o entendimento e a atividade dos alunos em nossos cursos. Um dos autores também é um desenvolvedor do sistema BlueJ.

Orientação a objetos com Java

Uma das razões para escolher o BlueJ foi que ele permite uma abordagem em que os professores lidam verdadeiramente com os conceitos importantes primeiro. 'Partindo de Objetos' foi um grito de batalha para muitos autores de livros didáticos e professores por algum tempo. Infelizmente, a linguagem Java não torna esse nobre objetivo muito fácil. Numerosos obstáculos de sintaxe e detalhes têm de ser sobrepujados antes de a primeira experiência com um objeto vivo surgir. O programa Java mínimo para criar e chamar um objeto em geral inclui:

- escrever uma classe;
- escrever um método main, incluindo conceitos como métodos estáticos, parâmetros e arrays na assinatura;
- uma instrução para criar o objeto ('new');
- uma atribuição a uma variável;
- a declaração de variável, incluindo tipo de variável;
- uma chamada de método, utilizando a notação de ponto;
- possivelmente uma lista de parâmetros.

Como resultado, os livros didáticos em geral

- têm de trabalhar à sua maneira nessa lista proibitiva e abordar apenas objetos em algum lugar no Capítulo 4; ou
- utilizar um programa no estilo 'Hello, world' com um único método main estático como o primeiro exemplo, não criando, assim, nenhum objeto.

Com o BlueJ, isso não é um problema. Um aluno pode criar um objeto e chamar seus métodos como uma primeira atividade! Como os usuários podem criar e interagir com objetos diretamente, conceitos como classes, objetos, métodos e parâmetros podem facilmente ser discutidos de maneira concreta antes de olhar para a primeira linha de sintaxe Java. Em vez de explicar mais sobre isso aqui, sugerimos que o leitor curioso mergulhe no Capítulo 1 – as coisas rapidamente se tornarão mais claras.

Uma abordagem iterativa

Outro aspecto importante deste livro é que ele segue um estilo iterativo. Na comunidade de ensino de informática, existe um padrão de projeto educacional bem conhecido que declara que conceitos importantes deveriam ser ensinados cedo e com freqüência.[1] É muito tentador para autores de livros didáticos tentar e dizer tudo sobre um tópico no ponto em que ele é introduzido. Por exemplo, é comum, ao introduzir tipos, fornecer uma lista completa de tipos de dados predefinidos ou discutir todos os tipos disponíveis de loop ao introduzir o conceito de loop.

Essas duas abordagens conflitam: não podemos nos concentrar na discussão de conceitos importantes primeiro e ao mesmo tempo fornecer cobertura completa de todos os tópicos encontrados. Nossa experiência com livros didáticos é que muitos detalhes inicialmente causam muita distração e têm o efeito de suprimir os pontos importantes, dificultando a compreensão.

Neste livro abordamos todos os tópicos importantes várias vezes, no mesmo capítulo e em capítulos diferentes. Os conceitos normalmente são introduzidos em um nível de detalhe necessário para entender e aplicar a tarefa à mão. São revisitados mais tarde em um contexto diferente, e a compreensão se aprofunda à medida que o leitor continua pelos capítulos. Essa abordagem também ajuda a lidar com a ocorrência freqüente de dependências mútuas entre conceitos.

Alguns professores talvez não conheçam uma abordagem iterativa. Olhando os primeiros capítulos, professores acostumados a uma introdução mais seqüencial se surpreenderão com o número de conceitos abordado tão cedo. Pode parecer uma curva de ensino íngreme.

É importante entender que esse não é o fim do assunto. Não se espera que os alunos entendam tudo sobre esses conceitos imediatamente. Em vez disso, esses conceitos fundamentais serão abordados repetidas vezes por todo o livro, permitindo que os alunos obtenham um entendimento cada vez mais profundo ao longo do tempo. Visto que seu nível de conhecimento muda à medida que o trabalho avança, revisitar tópicos importantes mais tarde permite que eles obtenham uma compreensão geral mais profunda.

Tentamos essa abordagem com alunos muitas vezes. Parece que os alunos têm menos problemas em lidar com essa abordagem que alguns professores mais experientes. E lembre-se: uma curva de ensino íngreme não é um problema contanto que você assegure que seus alunos podem escalá-la!

---

[1] O padrão 'Early Bird', em J. Bergin: 'Fourteen pedagogical patterns for teaching computer science', *Proceedings of the Fifth European Conference on Pattern Languages of Programs* (EuroPLop 2000), Irsee, Alemanha, julho de 2000.

Sem completa cobertura da linguagem

Relacionada com nossa abordagem iterativa está a decisão de não tentar fornecer cobertura completa da linguagem Java no livro.

O foco principal deste livro é transmitir princípios de programação orientada a objetos em geral, não os detalhes de linguagem Java em particular. Alunos que estudam com este livro podem trabalhar como profissionais de software pelos próximos 30 ou 40 anos de sua vida – é uma aposta relativamente segura de que a maioria de seu trabalho não será em Java. Todo livro didático importante deve naturalmente tentar prepará-los para algo mais fundamental do que a linguagem da moda.

Por outro lado, muitos detalhes do Java são importantes para realmente fazer trabalhos práticos.

Neste livro, abordamos construções Java em tantos detalhes quantos são necessários para ilustrar os conceitos à mão e implementar o trabalho prático. Algumas construções específicas para Java foram deliberadamente deixadas fora da discussão.

Estamos cientes de que alguns instrutores optarão por abordar alguns tópicos que não discutimos em detalhes. Isso é esperado e necessário. Entretanto, em vez de tentar cobrir cada possível tópico nós mesmos (e assim aumentar o tamanho deste livro para além de 1.500 páginas), lidamos com isso utilizando *ganchos*. Os ganchos são ponteiros, freqüentemente na forma de perguntas que levantam o tópico e oferecem referências para um apêndice ou um material externo. Esses ganchos asseguram que um tópico relevante é trazido à tona em um momento apropriado e permite ao leitor ou ao professor decidir com que nível de detalhe esse tópico deve ser abordado. Assim, os ganchos servem como um lembrete da existência do tópico e um marcador de lugar que indica um ponto na seqüência em que a discussão pode ser inserida.

Professores individualmente podem decidir por utilizar o livro tal como ele é, seguindo nossa seqüência sugerida ou adotar os desvios sugeridos pelos ganchos no texto.

Os capítulos também com freqüência incluem várias perguntas que sugerem o material de discussão relacionado ao tópico, mas não discutido neste livro. Esperamos que os professores discutam algumas dessas perguntas em classe ou que os alunos pesquisem as respostas como exercícios de dever de casa.

Abordagem baseada em projeto

A introdução de material no livro é baseada em projeto. O livro discute vários projetos de programação e fornece muitos exercícios. Em vez de introduzir uma nova construção e então fornecer um exercício para aplicar essa construção para resolver uma tarefa, primeiro fornecemos um objetivo e um problema. A análise do problema determina de que tipos de soluções precisamos. Como conseqüência, construções de linguagem são introduzidas à medida que são necessárias para resolver os problemas em questão.

Os capítulos iniciais fornecem pelo menos dois exemplos de discussão. Esses são projetos que são discutidos em detalhes para ilustrar conceitos importantes de cada capítulo. A utilização de dois exemplos muito diferentes suporta a abordagem iterativa: cada conceito é revisitado em um contexto diferente depois de ser introduzido.

Ao projetar este livro, tentamos utilizar um número grande e uma ampla variedade de projetos de exemplo diferentes. Esperamos que isso sirva para atrair o interesse do leitor, mas isso também ajuda a ilustrar a variedade de contextos diferentes em que os conceitos podem ser aplicados. É difícil encontrar bons projetos de exemplo. Esperamos que nossos projetos sirvam para dar aos professores bons pontos de partida e muitas idéias para uma ampla variedade de exercícios interessantes.

A implementação para todos os nossos projetos é escrita muito cuidadosamente, de modo que muitas questões periféricas podem ser estudadas lendo-se o código-fonte dos projetos. Acreditamos muito no benefício de aprender lendo e imitando bons exemplos. Para isso funcionar, porém,

alguém deve certificar-se de que os exemplos que os estudantes lêem são bem escritos e de que vale a pena imitá-los. Tentamos fazer isso.

Todos os projetos são desenvolvidos como problemas abertos. Embora uma ou mais versões de cada problema sejam discutidas em detalhe no livro, os projetos são desenvolvidos de modo que mais extensões e aperfeiçoamentos possam ser feitos como projetos de aluno. O código-fonte completo para todos os projetos vem incluído no livro. Uma lista de projetos discutida neste livro é fornecida na página xxv.

Seqüência dos conceitos em vez de construções de linguagem

Um outro aspecto que distingue este livro de muitos outros é que ele é estruturado ao longo de tarefas fundamentais de desenvolvimento de software e não necessariamente de acordo com as construções de linguagem Java particulares. Um indicador disso são os títulos de capítulo. Neste livro você não encontrará muitos dos tradicionais títulos de capítulo, como 'Tipos de dados primitivos' ou 'Estruturas de controle'. A estruturação por tarefas fundamentais de desenvolvimento permite-nos dar uma introdução muito mais geral que não é guiada pelas complexidades da linguagem de programação em particular utilizada. Também acreditamos que é mais fácil para o aluno seguir a motivação da introdução e que ela torna tudo muito mais interessante.

Como resultado dessa abordagem, não é tão simples utilizar este livro como um livro de referência. Livros didáticos introdutórios e livros de referência têm objetivos diferentes, parcialmente competitivos. Até certo grau, um livro didático pode tentar ser ambos, mas certos compromissos têm de ser feitos em alguns pontos. Nosso livro é claramente projetado como um livro didático e onde quer que um conflito tenha ocorrido, o estilo livro didático assumiu precedência sobre sua utilização como um livro de referência.

No entanto, fornecemos suporte para sua utilização como um livro de referência listando as construções Java apresentadas em cada capítulo na introdução do capítulo.

Seqüência dos capítulos

O Capítulo 1 lida com os conceitos mais fundamentais da orientação a objetos: objetos, classes e métodos. É dada uma introdução prática e sólida a esses conceitos sem entrar nos detalhes da sintaxe Java. Também se faz um primeiro exame de alguns códigos-fonte. Fazemos isso utilizando um exemplo de formas gráficas que interativamente pode ser desenhado e um segundo exemplo de um sistema simples de matrículas dos alunos em aulas de laboratório.

O Capítulo 2 esclarece definições de classe e investiga como o código-fonte Java é escrito para criar comportamento de objetos. Discutimos como definir campos e implementar métodos. Aqui, também introduzimos os primeiros tipos de instrução. O principal exemplo é uma implementação de uma máquina de vender bilhetes. Também voltamos ao exemplo de aula de laboratório do Capítulo 1 para investigar isso um pouco mais.

O Capítulo 3 então expande o quadro para discutir interação de múltiplos objetos. Vemos como os objetos podem colaborar invocando os métodos uns dos outros para realizar uma tarefa comum. Também discutimos como um objeto pode criar outros objetos. Discutimos um relógio digital, que utiliza dois objetos de exibição de número para mostrar horas e minutos. Como um segundo exemplo importante, examinamos uma simulação de um sistema de correio eletrônico em que as mensagens podem ser enviadas entre clientes de correio.

No Capítulo 4 continuamos construindo estruturas mais extensas de objetos. Acima de tudo, começamos a utilizar coleções de objetos. Implementamos um bloco de notas eletrônico e um sis-

tema de leilão para introduzir coleções. Ao mesmo tempo, discutimos a iteração por coleções e verificamos inicialmente os loops. A primeira coleção utilizada é um `ArrayList`. Na segunda metade do capítulo introduzimos arrays como uma forma especial de uma coleção e o loop for como outra forma de um loop. Discutimos uma implementação de um analisador de log Web como um exemplo para utilização de arrays.

O Capítulo 5 lida com bibliotecas e interfaces. Introduzimos a biblioteca Java padrão e discutimos algumas classes de biblioteca importantes. Sobretudo, explicamos como ler e entender documentação de bibliotecas. A importância da documentação escrita em projetos de desenvolvimento de software é discutida e acabamos por praticar como escrever documentação adequada para as próprias classes. `Random`, `Set` e `Map` são exemplos de classes que encontramos neste capítulo. Implementamos um sistema de diálogo semelhante ao *Eliza* e uma simulação gráfica de uma bola quicando para aplicar essas classes.

O Capítulo 6, intitulado *Objetos bem-comportados*, trata de um grupo inteiro de questões relacionadas à produção de classes corretas, compreensíveis e sustentáveis. Ele aborda questões como escrever código claro e compreensível – incluindo estilos e comentários – para teste e depuração. As estratégias de teste são introduzidas e vários métodos de depuração são discutidos em detalhes. Utilizamos um exemplo de um diário para agendamento de compromisso e uma implementação de uma calculadora eletrônica para discutir esses tópicos.

No Capítulo 7, discutimos mais formalmente as questões de dividir o domínio de um problema em classes para implementação. Introduzimos questões para projetar classes bem, incluindo conceitos como projeto baseado em responsabilidade, acoplamento, coesão e refatoração. Um jogo de aventura interativo baseado em texto (*World-of-Zuul*) é utilizado para essa discussão. Passamos por várias iterações em aperfeiçoar a estrutura interna de classe do jogo e estender sua funcionalidade e terminamos com uma longa lista de propostas para extensões que podem ser feitas como projetos de aluno.

Os Capítulos 8 e 9 introduzem herança e polimorfismo, estudando detalhadamente muitas das questões relacionadas. Discutimos um banco de dados simples de CDs e vídeos para ilustrar os conceitos. Questões de herança de código, subtipagem, chamadas de método polimórficos e cancelamento são discutidas em detalhe.

No Capítulo 10 implementamos uma simulação de caçador/caça. Essa simulação serve para discutir mecanismos adicionais de abstração baseados em herança, isso é, interfaces e classes abstratas.

O Capítulo 11 então aborda a difícil questão de como lidar com erros. Vários problemas e soluções possíveis são discutidos e o mecanismo de tratamento de exceção do Java é discutido em detalhe. Estendemos e aprimoramos uma aplicação de catálogo de endereços para ilustrar os conceitos.

O Capítulo 12 volta a discutir com mais detalhes o próximo nível de abstração: como estruturar um problema vagamente descrito em classes e métodos. Em capítulos anteriores assumimos que partes grandes da estrutura da aplicação já existem e fizemos aprimoramentos. Agora é hora de discutir como podemos começar a partir de uma lousa limpa. Isso envolve discussão detalhada sobre o que as classes devem ser para implementar nossa aplicação, como eles interagem e como as responsabilidades devem ser distribuídas. Utilizamos cartões Classe/Responsabilidades/Colaboradores (CRC) para abordar esse problema, durante o projeto de um sistema de reserva de lugares em um cinema.

No Capítulo 13, tentamos reunir tudo e integrar muitos tópicos a partir de capítulos anteriores do livro. É um estudo de caso completo, começando com o projeto da aplicação, passando pelo projeto de interfaces de classe, até a discussão de muitas características funcionais e não-funcionais importantes e detalhes de implementação. Os tópicos discutidos em capítulos anteriores (como confiabilidade, estruturas de dados, projeto de classe, teste e extensibilidade) são aplicados novamente em um novo contexto.

Grupo de discussão

Os autores mantêm um ativo grupo de discussão por correio eletrônico com a finalidade de facilitar troca de idéias e oferecer suporte mútuo para e por leitores deste livro e outros usuários do BlueJ. As mensagens nessa lista são arquivadas e disponibilizadas para acesso público. Utilizando essa lista, os professores podem receber suporte e opiniões de outros professores e dos autores deste livro. O endereço para essa lista é `bluej-discuss@bluej.org`. Pessoas interessadas podem ingressar na lista ou navegar pelos repositórios de arquivos em

```
http://lists.bluej.org/mailman/listinfo/bluej-discuss
```

Material adicional

Este livro inclui todos os projetos utilizados como exemplos de discussão e exercícios em um CD. Para facilitar a compreensão, traduzimos os comentários dos programas no livro, mas no CD os programas estão exatamente como foram criados pelos autores, com os comentários em inglês. O CD também inclui o ambiente Java de desenvolvimento (Java Development Environment – JDK) e o BlueJ para vários sistemas operacionais.

Há um site Web de suporte para este livro em

```
http://www.bluej.org/objects-first
```

Nesse site Web, podem ser encontradas atualizações para os exemplos e é fornecido material adicional (em inglês). Por exemplo, o guia de estilos utilizado para todos os exemplos neste livro está disponível na forma eletrônica, de modo que instrutores podem modificá-lo para atender a seus próprios requisitos. Inclui também uma seção, protegida por senha, apenas para professores. Essa seção fornece material adicional, além de um conjunto de slides para ministrar um curso com este livro.

Companion Web site

O site Web do livro em `http://www.prenhall.com/barnes_br` contém a tradução das apresentações do PowerPoint e recursos para o professor (em inglês).