

Padrões JavaScript

Stoyan Stefanov

Authorized Portuguese translation of the English edition of titled *JavaScript Patterns, First Edition*, ISBN: 978-0-596-80675-0 © 2010, Stoyan Stefanov. This translation is published and sold by permission of O'Reilly Media, Inc., the owner of all rights to publish and sell the same.

Tradução em português autorizada da edição em inglês do título *JavaScript Patterns, First Edition*, ISBN: 978-0-596-80675-0 © 2010, Stoyan Stefanov. Esta tradução é publicada e vendida com a permissão da O'Reilly Media, Inc., detentora de todos os direitos para publicação e venda desta obra.

© Novatec Editora Ltda. 2011.

Todos os direitos reservados e protegidos pela Lei 9610 de 19/02/1998. É proibida a reprodução desta obra, mesmo parcial, por qualquer processo, sem prévia autorização, por escrito, do autor e da Editora.

Editor: Rubens Prates
Tradução: Edgard Damiani
Revisão gramatical: Carla Mello Moreira
Editoração eletrônica: Camila Kuwabata e Carolina Kuwabata

ISBN: 978-85-7522-266-9

Histórico de impressões:

Janeiro/2011 Primeira edição

Novatec Editora Ltda.
Rua Luís Antônio dos Santos 110
02460-000 – São Paulo, SP – Brasil
Tel.: +55 11 2959-6529
Fax: +55 11 2950-8869
Email: novatec@novatec.com.br
Site: www.novatec.com.br
Twitter: twitter.com/novateceditora
Facebook: facebook.com/novatec
LinkedIn: linkedin.com/in/novatec

**Dados Internacionais de Catalogação na Publicação (CIP)
(Câmara Brasileira do Livro, SP, Brasil)**

Stefanov, Stoyan
Padrões JavaScript / Stoyan Stefanov ;
[tradução Edgard Damiani]. -- São Paulo :
Novatec Editora ; Sebastopol, CA. : O'Reilly,
2010.

Título original: JavaScript patterns.
ISBN 978-85-7522-266-9

1. JavaScript (Linguagem de programação para
computadores) I. Título.

10-13324

CDD-005.133

Índices para catálogo sistemático:

1. JavaScript : Linguagem de programação :
Computadores : Processamento de dados
005.133
OGF_20101207

Sumário

| | |
|---|-----------|
| Prefácio | 13 |
| Capítulo 1 ■ Introdução | 17 |
| Padrões | 17 |
| JavaScript: conceitos | 19 |
| Orientado a objeto..... | 19 |
| Sem classes | 20 |
| Protótipos..... | 20 |
| Ambiente | 21 |
| ECMAScript 5 | 21 |
| JSLint | 22 |
| Objeto console | 23 |
| Capítulo 2 ■ Conceitos básicos..... | 25 |
| Escrevendo código manutenível | 25 |
| Minimizando globais..... | 26 |
| O problema das variáveis globais..... | 27 |
| Efeitos colaterais de quando se esquece var | 28 |
| Acessando o objeto global | 29 |
| Padrão var único (single var) | 30 |
| Hoisting: o problema das declarações var espalhadas | 31 |
| Loops for | 32 |
| Loops for-in | 34 |
| (Não) expandir protótipos embutidos..... | 36 |
| Padrão switch..... | 37 |
| Evitar conversões de tipo implícitas..... | 38 |
| Evite eval() | 38 |
| Conversão de números com parseInt()..... | 40 |
| Convenções de programação..... | 41 |
| Indentação | 41 |
| Chaves | 42 |
| Localização da chave de abertura | 43 |
| Espaços em branco | 44 |
| Convenções de nomeação | 45 |
| Construtores iniciados com maiúsculas | 46 |
| Separação de palavras | 46 |
| Outros padrões de nomeação | 46 |
| Escrevendo comentários | 48 |

| | |
|--|-----------|
| Escrevendo documentos de API..... | 48 |
| Exemplo no YUIDoc..... | 49 |
| Escrevendo para ser lido..... | 53 |
| Revisões por pares..... | 54 |
| Minificando (n)a produção..... | 54 |
| Execute o JSLint..... | 55 |
| Resumo..... | 56 |
| Capítulo 3 ■ Literais e construtores..... | 57 |
| Objeto literal..... | 57 |
| Sintaxe de objeto literal..... | 58 |
| Objetos a partir de um construtor..... | 59 |
| Pegadinha do construtor Object..... | 60 |
| Funções construtoras customizadas..... | 60 |
| Valores de retorno dos construtores..... | 62 |
| Padrões que impõem o uso de new..... | 63 |
| Convenção de nomeação..... | 63 |
| Usando that..... | 64 |
| Construtor autoinvocável..... | 64 |
| Array literal..... | 65 |
| Sintaxe de array literal..... | 66 |
| Curiosidades do construtor Array()..... | 66 |
| Verificando a “natureza” de um array..... | 67 |
| JSON..... | 68 |
| Trabalhando com o JSON..... | 68 |
| Expressão regular literal..... | 69 |
| Sintaxe literal de expressão regular..... | 70 |
| Encapsuladores primitivos..... | 71 |
| Objetos de erro..... | 72 |
| Resumo..... | 73 |
| Capítulo 4 ■ Funções..... | 75 |
| Conceitos básicos..... | 75 |
| Esclarecimentos sobre a terminologia..... | 76 |
| Declarações versus expressões: nomes e hoisting..... | 78 |
| Propriedade name da função..... | 78 |
| Hoisting de função..... | 79 |
| Padrão callback..... | 81 |
| Um exemplo de callback..... | 81 |
| Callbacks e escopo..... | 83 |
| Listeners de eventos assíncronos..... | 85 |
| Intervalos (timeouts)..... | 85 |
| Callbacks em bibliotecas..... | 86 |
| Retornando funções..... | 86 |
| Funções autodefiníveis..... | 87 |

| | |
|---|------------|
| Funções imediatas | 88 |
| Parâmetros de uma função imediata..... | 89 |
| Valores retornados de funções imediatas | 90 |
| Benefícios e utilização | 91 |
| Inicialização de um objeto imediato | 92 |
| Ramificação em tempo de inicialização | 94 |
| Propriedades de função – Um padrão de memoização..... | 96 |
| Objetos de configuração | 97 |
| Curry..... | 99 |
| Aplicação de função..... | 99 |
| Aplicação parcial..... | 100 |
| Currying..... | 101 |
| Quando usar o currying..... | 104 |
| Resumo..... | 104 |
| Capítulo 5 ■ Padrões de criação de objetos | 106 |
| Padrão de namespace..... | 106 |
| Função-namespace de propósito geral..... | 108 |
| Declarando dependências | 109 |
| Propriedades e métodos privados | 111 |
| Membros privados..... | 111 |
| Métodos privilegiados..... | 112 |
| Falhas na privacidade..... | 112 |
| Objetos literais e privacidade..... | 114 |
| Protótipos e privacidade..... | 115 |
| Revelando funções privadas como métodos públicos..... | 115 |
| Padrão de módulo | 117 |
| Padrão de módulo de revelação | 119 |
| Módulos que criam construtores | 119 |
| Importando globais para dentro de um módulo..... | 120 |
| Padrão de sandbox | 121 |
| Um construtor global..... | 121 |
| Adicionando módulos..... | 123 |
| Implementando o construtor..... | 124 |
| Membros estáticos..... | 125 |
| Membros estáticos públicos | 126 |
| Membros estáticos privados | 127 |
| Constantes de objeto | 129 |
| Padrão de encadeamento | 131 |
| Prós e contras do padrão de encadeamento..... | 132 |
| Método method() | 132 |
| Resumo..... | 134 |
| Capítulo 6 ■ Padrões de reutilização de código | 135 |
| Clássico versus moderno: padrões de herança..... | 135 |
| Resultado esperado quando se usa herança clássica | 136 |

| | |
|---|------------|
| Padrão clássico 1 – padrão comum | 137 |
| Seguindo a cadeia de protótipos | 137 |
| Problemas no uso do padrão 1..... | 140 |
| Padrão clássico 2 – Rent-a-Constructor (aluga-se um construtor) | 140 |
| A cadeia de protótipos..... | 141 |
| Herança múltipla por empréstimo de construtores | 142 |
| Prós e contras do padrão de empréstimo de construtor..... | 143 |
| Padrão clássico 3 – Aluga e define (rent and set) o protótipo..... | 143 |
| Padrão clássico 4 – Compartilhe o protótipo..... | 145 |
| Padrão clássico 5 – um construtor temporário | 146 |
| Armazenando a superclasse..... | 147 |
| Redefinindo o ponteiro do construtor | 147 |
| Klass..... | 149 |
| Herança prototípica | 151 |
| Discussão | 153 |
| Inclusão na ECMAScript 5 | 154 |
| Herança por cópia de propriedades | 154 |
| Mix-ins..... | 156 |
| Métodos emprestados..... | 157 |
| Exemplo: emprestando de um array | 158 |
| Empresta e liga (borrow and bind) | 158 |
| Function.prototype.bind() | 160 |
| Resumo..... | 161 |
| Capítulo 7 ■ Padrões de projeto | 162 |
| Singleton..... | 162 |
| Usando new | 163 |
| Instância em uma propriedade estática | 164 |
| Instância em um closure | 164 |
| Fábrica (Factory) | 167 |
| Fábrica de objetos embutida..... | 169 |
| Iterador (Iterator) | 170 |
| Decorador (Decorator) | 172 |
| Utilização..... | 172 |
| Implementação..... | 173 |
| Implementação usando uma lista | 175 |
| Estratégia (Strategy) | 177 |
| Exemplo de validação de dados..... | 177 |
| Fachada (Façade) | 180 |
| Proxy | 182 |
| Um exemplo..... | 182 |
| Proxy como cache | 189 |
| Mediador (Mediator) | 190 |
| Exemplo de mediador..... | 191 |

| | |
|--|------------|
| Observador (Observer) | 193 |
| Exemplo 1: Assinatura de revista | 194 |
| Exemplo 2: o jogo de pressionamento de teclas | 197 |
| Resumo..... | 201 |
| Capítulo 8 ■ DOM e padrões de navegador..... | 203 |
| Separação de interesses (separation of concerns) | 203 |
| Script em DOM..... | 205 |
| Acesso ao DOM..... | 205 |
| Manipulação do DOM..... | 206 |
| Eventos | 208 |
| Tratamento de eventos | 208 |
| Delegação de eventos | 210 |
| Scripts de execução longa..... | 212 |
| setTimeout()..... | 212 |
| Web workers..... | 212 |
| Script remoto | 213 |
| XMLHttpRequest | 213 |
| JSONP | 215 |
| Frames e sinalizadores de imagens..... | 218 |
| Implantando JavaScript | 219 |
| Combinando scripts | 219 |
| Minificando e comprimindo | 220 |
| Cabeçalho Expires | 220 |
| Usando uma CDN..... | 221 |
| Estratégias de carregamento..... | 221 |
| Lugar do elemento <script> | 222 |
| HTTP em bloco..... | 223 |
| Elemento <script> dinâmico para downloads não bloqueadores | 225 |
| Carregamento preguiçoso..... | 227 |
| Carregamento sob demanda..... | 227 |
| Precarregando o JavaScript | 229 |
| Resumo..... | 231 |
| Índice remissivo | 232 |